VYSOKÁ ŠKOLA BÁŇSKÁ – TECHNICKÁ UNIVERZITA OSTRAVA
EKONOMICKÁ FAKULTA

KATEDRA APLIKOVANÉ INFORMATIKY

Návrh databázové aplikace pro sportovní klub

Design of Database Application for Sports Club

Student: Tomáš Janků

Vedoucí bakalářské práce: Ing. Vítězslav Novák, Ph.D.

Ostrava 2014

VŠB - Technická univerzita Ostrava
Ekonomická fakulta
Katedra aplikované informatiky

Zadání bakalářské práce

Student: **Tomáš Janků**

Studijní program: B6209 Systémové inženýrství a informatika

Studijní obor: 6209R001 Aplikovaná informatika

Téma: Návrh databázové aplikace pro sportovní klub
Design of Database Application for Sports Club

Zásady pro vypracování:

1. Úvod
2. Teoretická východiska
3. Analýza současného stavu
4. Návrh a implementace databázové aplikace
5. Závěr

Seznam použité literatury
Seznam zkratek
Prohlášení o využití výsledků bakalářské práce
Seznam příloh
Přílohy

Seznam doporučené odborné literatury:

PÍSEK, Slavoj. *Access 2010: podrobný průvodce.* Praha: Grada, 2011. ISBN 978-80-247-3653-2.
KRUCZEK, Aleš. *1001 Tipů a triků pro Microsoft Access 2007/2010.* Brno: Computer Press, 2011. ISBN 978-80-251-3507-5.
SHEPHERD, Richard. *Access VBA: výukový kurz.* Přeložil Jakub MUŽÍK. Brno: Computer Press, 2012. ISBN 978-80-251-3686-7.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Vítězslav Novák, Ph.D.**

Datum zadání: 22.11.2013
Datum odevzdání: 09.05.2014

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA * OSTRAVA * Ekonomická fakulta 1

Ing. Petr Rozehnal, Ph.D.
vedoucí katedry

prof. Dr. Ing. Dana Dluhošová
děkanka fakulty

Místopřísežné prohlášení o samostatném vypracování bakalářské práce

Prohlašuji, že jsem celou práci, včetně příloh, vypracoval samostatně.

Podpis: 

Datum odevzdání práce: 9. 5. 2014

Touto cestou bych chtěl poděkovat svému vedoucímu bakalářské práce Ing. Vítězslavu Novákovi, Ph.D. za jeho cenné rady, připomínky, které mi velmi pomohly k vypracování této práce.

OBSAH

1 Úvod

Neustálý posun kupředu ve světě informačních technologií je důkazem zainteresovanosti těchto technologií takovým způsobem, že pro většinu populace se stávají neodmyslitelnou součástí každodenního života. Dopomáhají uživatelům k usnadnění práce, ušetření času, zlepšení přehlednosti, ale také ke strávení volna či usnadnění komunikace s okolním světem. Právě s tématy usnadnění práce a zlepšení přehlednosti je mimo jiné spjata i tato bakalářská práce, která má dopomoci ke zlepšení stávající situace vedení dat ve sportovním klubu.

Tématem, kterým se tato práce zabývá, je vytvoření databázové aplikace pro sportovní klub, konkrétně pro klub házenkářský. Databázová aplikace bude sloužit především sekretariátu, trenérům a vedoucím mužstev k přehledu o hráčích, jejich statistikách v jednotlivých zápasech, přehledu o klubové oblečení apod.

Cíle bakalářské práce jsou následovné:

- sběr a zaznamenání osobních informací o jednotlivých hráčích,
- vytvoření databázové aplikace pro sportovní klub, která by nahradila kartotékové zpracování dat v kombinaci s tabulkovým procesorem MS Excel,
- zlepšení informovanosti a přehledu o výkonech jednotlivých hráčů, jak pro vedení klubu, tak i pro trenéry v následném hodnocení.

Databázová aplikace bude určena pro sekretariát klubu, kam jednotliví trenéři a vedoucí budou dodávat data, která budou do aplikace vkládána. Tyto data poté budou k dispozici jak trenérům při hodnocení jednotlivých zápasů či hodnocení určitého bloku zápasů, ale i pro přehled aktuálního stavu družstva, výkonnosti jednotlivých hráčů nebo pro následnou komunikaci s médii. Aplikace by měla splňovat základní požadavky potřebné pro zmíněné akce jako je například přidání, vymazání či úprava hráče konkrétního týmu, zařazení hráče do konkrétního zápasu, prezentace statistik z jednotlivých zápasů pomocí sestav apod.

Koncept této práce je členěn do pěti základních kapitol. V první kapitole je v rámci úvodu seznámení s tématem práce, stanovení cílů práce a popis základní problematiky.

V následující kapitole *„Teoretická východiska“* je popisován teoretický základ k problematice databází, nástrojů a pojmů s nimi souvisejícími. Následně zde bude seznámeno s programem Microsoft Access a jeho nástroji, včetně programovacího jazyk Visual Basic for Application (VBA).

Třetí kapitolou je *„Analýza současného stavu“*, kde je popisována momentální situace

zpracovávání dat ve sportovním klubu.

Zásadní kapitolou je v pořadí čtvrtá, a to *„Návrh a implementace databázové aplikace“*. Je zaměřena na samotný návrh aplikace v prostředí Microsoft Access, a to za pomocí tabulkového návrhu, propojení těchto tabulek relacemi, převedení tabulek do formulářů či dalších objektů a s využitím procedur jazyka VBA dokončení aplikace do finálního grafického návrhu.

Poslední, pátou kapitolou je *„Závěr“*, kde přijde na řadu celkové hodnocení práce, dosažení stanovených cílů, či zdůvodnění nesplnění stanovených cílů.

2 Teoretická východiska

V běžném životě je pravidlem, že s každým řešením určité problematiky, ať už každodenních úkolů či řešení složitých projektů je zakládáno na znalostech. Bylo nepřípustné, aby tomu bylo jinak i v rámci této bakalářské práce. Proto i pro vytvoření databázové aplikace je potřebné získat teoretický základ, jenž je zahrnut v rámci této kapitoly.

2.1 Zpracování dat

Na první pohled je zřejmá spojitost mezi daty a databází. Databáze je považována za strukturované datové úložiště, kam jsou vkládána data. Po zpracování, úpravách a přidání souvislostí, jsou následně vytvářeny patřičné výsledky, které mohou dopomoci k analýzám, rozhodování či vyhledávání a kategorizaci informací. [6]

2.2 Definice databáze

Velká představivost lze uplatnit při seznámení s pojmem databáze. Tento pojem není čistě IT záležitostí. I pouhé zaznamenávání informací na papír a vkládání do složek bylo, a stále je, databází.

Z tohoto tedy vyplývá, že databáze souvisí s uchováváním dat na paměťové médium, kde při ukládání nezáleží, jak je dané médium strukturováno. [10]

Databáze je tedy označována jako úložiště dat, kde jsou v rámci kompaktní jednoty data zpracovávána na základě jejich kontextové souvislosti a popisu. [5]

2.3 Systém řízení báze dat

Systém řízení báze dat (SŘBD), vycházející z anglického názvu Database Management Systems (DBSM), spojuje uživatele, databázovou aplikaci a databázi. Představuje tak rozhraní zajišťující uživateli služby jako definice, návrh, údržba a správa databáze mezi aplikačními programy a databází. [1] [5]

Mezi základní služby SŘBD patří například:

- přesouvání velkého množství dat na fyzické datové úložiště nebo jejich získávání,
- zálohovací mechanismy při případných haváriích,
- podpora dotazů při načítání dat, pomocí příslušného dotazovacího jazyka (například SQL),

- zajištění integrity databáze[1].

Jedničkou v této oblasti jsou softwarové produkty od společnosti Oracle, následně produkty od IBM (DB2) a v neposlední řadě produkty společnosti Microsoft (Microsoft Access, Microsoft SQL Server). [5]

2.4 Databázové modely

Ve světě databází jsou různé druhy databázových modelů. Od nejstarších modelů zakládajících hierarchii až po datové modely pomocí relací.

2.4.1 Hierarchický databázový model

Hierarchický databázový model představuje první podobu databáze. Je založen na struktuře nadřízený-podřízený (nebo též rodič-potomek). Základem je kořenová tabulka, která se nachází na vrcholu hierarchie a z tohoto kořene vycházejí podřízené tabulky, nazývány též větve. Takováto struktura znamená, že jedna nadřízená tabulka smí mít několik podřízených, ale každá podřízená tabulka smí mít právě jednu nadřízenou. Propojenost tabulek je zajištěna pomocí ukazatelů, které dávají přesnou informaci, kde se tabulka nachází. [5] [2]

Explicitní propojení se staví do popředí jako jedna z největších výhod tohoto modelu, a to především díky vysoké rychlosti vyhledávání dat. Následnou výhodou je automatické vymazání záznamu v tabulce podřízeného, pokud v nadřízené tabulce je vymazán záznam propojen právě s tabulkou podřízeného. Tato vlastnost je nazývána referenční integrita.

Co však nestaví do popředí, ale na konec, tj. do nevýhod, je vkládání dalších záznamů do hierarchie. Tento problém vzniká, pokud by bylo nutné přidat dalšího podřízeného například s jinými záznamy v tabulce, ale tento podřízený, musí mít stejného nadřízeného, což podle pravidla hierarchie není možné. Neméně významným problémem je vytvoření stejných záznamů v různých tabulkách z důvodu pevně dané struktury, tzv. redundantních dat. [2]

2.4.2 Síťový databázový model

Odstranit nedostatky a problémy modelu hierarchického bylo hlavní vizí síťového modelu. Model je založena na základě uzlů a množinových struktur. Uzel je definován jako souhrn záznamů, kde vztahy mezi uzly zajišťuje množinová struktura propojení. Jako v hierarchickém modelu bylo využíváno struktury nadřízený-podřízený, zde je využívána vlastník-člen. [2]

Model je založen na vztahovém principu jedna ku více, kde záznam ve vlastníkově

[1] Integrita databáze – ověřování správnosti uložených dat

uzlu smí být ve vztahu k více záznamům v uzlu člen. V opačném případě, záznam uzlu člen je přidružen pouze k jednomu záznamu v uzlu vlastník. [5]

Musí zde být dodrženo i pravidlo, že záznam v uzlu člen nesmí figurovat bez vztahu k záznamu v uzlu vlastník, kdežto v opačném pořadí to lze.

Jak již bylo nastíněno, vztahy jsou zajištěny pomocí množinových struktur. Princip množin je: *„Mezi dvěma uzly může být definována jedna nebo více množin (spojení), a libovolný uzel může být součástí dalších množin s jinými uzly v databázi“*. Pomocí těchto množinových struktur je možný přístup k databázi z libovolné množinové struktury či uzlu, což je jednou z hlavních výhod oproti hierarchickému modelu. Možnost vytváření souhrnnějších dotazů v databázi, a tím zrychlení vyhledávání dat je dalším zlepšením oproti modelu hierarchickému.

Čeho však síťový model nedosáhnul je celkové zjednodušení pro uživatele. Je totiž nezbytně nutné, aby uživatel měl dokonalý přehled o struktuře databáze a důkladně tuto strukturu znal. V neposlední řadě je velmi obtížné strukturu změnit, a to z důvodu, že různé aplikace využívají tuto strukturu a odkazují se na ní. [2]

2.4.3 Relační databázový model

Model relačních databází je vyústěním snahy vědců odstranit složité vyhledávací procesy v pevné struktuře předchozích modelů. Základní myšlenkou je nahradit předem, přesně, složitě, zdlouhavě a tudíž neprakticky definovanou vyhledávací cestu efektivnějším způsobem vyhledávání dat. [5]

Hlavními prvky relačních databází jsou dvourozměrné tabulky. Data jsou ukládána do těchto tabulek, kde jsou na základě logických vztahů propojeny s dalšími tabulkami. Tyto vztahy jsou nazývány relace. Uspořádání dat v tabulce je pomocí tzv. n-tic (záznamů), které představují řádek tabulky a atributů (polí), které představují sloupec tabulky. Na pořadí záznamů v těchto tabulkách není brán zřetel, jako tomu bylo u předchozích modelů, protože každý záznam má v rámci svého pole identifikátor tzv. primární klíč, který záznamu dodává jedinečnost. O záznamy na fyzickém médiu se v tomto modelu již stará SŘBD, tudíž je uživatel zbaven starosti ohledně zapamatování si fyzického umístění dat. [2] [10]

Jak již bylo mírně nastíněno, tak hlavním krokem kupředu oproti předchozím modelům je zrychlení a zefektivnění vyhledávacích procesů. Další výhodou je, že již nedochází k tak častým chybám, protože tabulky již nemusí obsahovat kompletní informace z propojené tabulky. Pokud se chyba náhodou objeví (například překlepnutí u zadávání jména), tak nebude mít vliv na spolehlivost databáze, jak tomu bylo v předešlých modelech a

lze ji snadno opravit a oprava se projeví do všech souvisejících záznamů. [6]

S výhodou vyhledávacích procesů je spjato i tzv. spojování tabulek, které umožňuje komplexní dotazování z více tabulek v rámci jednoho dotazu.

Relační model je nejvyužívanější model, ale i tak jsou zde nalezeny některé nevýhody. Je to především neschopnost pracovat se složitějšími datovými typy, jako jsou obrázky, audio a video soubory. [5]

2.4.4 Objektově orientovaný model

Podkladem zde je sada programovacích, objektově orientovaných jazyků (například Java). Model pracuje na principu tzv. zapouzdření objektů. Objekt si lze představit jako soubor dat, která jsou v příbuzném vztahu a programové logiky (například zákazník). Ke každému objektu jsou přiřazeny jednotlivé datové položky nazývány jako proměnné. (například jméno zákazníka). K těmto proměnným lze přistupovat pouze prostřednictvím tzv. metod, které provádí nad objekty příslušné operace (například přidání nového zákazníka). Přistupování k proměnným pouze pomocí metod je již zmíněná zapouzdřenost objektu.

Tento model se však příliš neuchytil z důvodu absence vytváření souhrnných dotazů, a tak byl využit jen pro zpracování složitějších souborů. [5]

2.4.5 Objektově relační model

Vizí ve zpracování rozsáhlejších datových typů v relačních databázích je právě tento model. Je pojat jako kombinace objektově orientovaného a relačního modelu, kde kombinací a převzetím výhod těchto dvou modelů se snaží o odstranění nedostatků samostatného relačního modelu přidáním objektově orientovaných funkcí. [2]

2.5 Databázové pojmy

Databáze logicky souvisí s daty, takže je na místě zmínit jejich definici. Data jsou neupravenými údaji, které jsou uchovávány v databázi. Pojmem data je nazýváno to, co je do databáze ukládáno. Pokud je datům přiřazen patřičný význam, tak vznikají informace.

Informace jsou definovány jako zpracovaná data, která jsou smysluplná, srozumitelná a mají patřičný význam. Informace představuje to, co je z databáze získáváno. Definici si zaslouží i specifická hodnota záznamu, a to hodnota NULL.

NULL je hodnota záznamu, která nereprezentuje nulu. Tato hodnota je buďto záměrně či omylně chybějící nebo neznámá. [2]

2.5.1 Tabulka

O tabulkách již bylo drobně zmíněno při vysvětlování relačního modelu. Nicméně pro upřesnění definice je tabulka hlavní strukturou v relačních databázích. Je tvořena záznamy (tzv. n-ticemi) a poli (tzv. atributy), kde jejich význam uspořádání v tabulce je nulový. Co však má obrovský význam je tzv. primární klíč, který dává každému záznamu jedinečnou hodnotu (viz definice primárního klíče podpodkapitola 2.6.3). Pole či skupina polí tvořících primární klíč je zásadní jak pro identifikaci samotného pole (skupiny polí), tak hlavně pro navázání vztahu s další tabulkou.

Základní rozdělení tabulek je dle dvou kritérií. Je to podle způsobu ukládání a obsahu dat. Podle obsahu se tabulky dělí:

1) **Objektové** - reprezentují hmatatelné objekty (například pacient u lékaře).
2) **Událostní** – reprezentují události v čase, které je potřeba zaznamenat (například návštěva u lékaře).

Druhým způsobem dělení je dle způsobu ukládání:

1) **Datová tabulka** – je nejčastějším typem a má za úkol ukládat data, která jsou zde zpracovávána na informace.
2) **Kontingenční tabulka** – tabulka, kde jsou především často neměnná data (například kódy produktů) a využívají se především při kontrole integrity dat. [2]

2.5.2 Pohled

Pohledem je virtuální tabulka, skládající se z jednoho či více polí základních tabulek databáze. V obvyklých případech je pohled vytvářen pomocí tzv. dotazů. V rámci pohledů lze tedy docílit i spojení tabulek.

Hlavní výhodou pohledů je poskytnutí širokého úhlu pozorování informací v databázi s možností obrovské flexibility. Následně se využívá z bezpečnostního hlediska pro skrytí dat, které určití uživatelé nemusí vidět. [1]

2.5.3 Klíče

O jednom z klíčů bylo již dříve zmiňováno v problematice relací databáze neboli také propojení tabulek. Bylo zmíněno, že každá tabulka musí v relační databázi obsahovat pole či skupinu polí definovanou jako primárním klíčem.

Primární klíč však není jediným klíčem, a tak je obecně pojem klíč definován jako

specifické pole, založené na logické struktuře, které má významnou úlohou v rámci vytváření tabulkových relací. [2]

Jednotlivé typy klíčů jsou:

1) **Super klíč** – definice říká, že *„je to sloupec či množina sloupců, které jedinečně identifikují záznam v relaci“*. [1]
2) **Kandidátní klíč** – vychází z definice super klíče, ale k jedinečné identifikaci záznamů využívá maximální korekci potřebných sloupců (příkladem kandidátních klíčů jsou například identifikační číslo zaměstnance či rodné číslo zaměstnance).
3) **Primární klíč –** vychází z definice kandidátního klíče, takže opět musí splňovat vlastnost jedinečné identifikace jakéhokoliv záznamu v tabulce. Definice primárního klíče tak zní: *„Kandidátní klíč, který je vybrán, aby jedinečně určoval záznamy v tabulce.“* (příkladem primárního klíče je rodné číslo zaměstnance).
4) **Alternativní klíč –** vychází opět z definice kandidátního klíče, ale s tím rozdílem, že nebyl vybrán jako primární klíč (příkladem je neučiněný výběr identifikačního čísla zaměstnance). [1] [2]
5) **Cizí klíč –** je kopií primárního klíče při vytváření relačních vztahů mezi tabulkami. Principiálně se kopie primárního klíče jedné tabulky přidá jako pole druhé tabulky, kde se poté v rámci příslušného relačního vztahu tyto tabulky propojí. [2]

2.5.4 Indexy

Indexy jsou informace o řazení záznamů. Ukládání indexů je v rámci tabulky, jako její vlastnost. Jejich základní podstatou je rychlejší vyhledávání na základě setříděných dat, které index reprezentuje.

Využití je především pro pole, podle kterých se budou data řadit (typickým příkladem je jméno či příjmení) či pole, která souvisejí s relací. Umožňují takové vlastnosti jako zakázání duplicitních hodnot, řazení na základě dvou či více polí (například při shodě příjmení je využito následujícího pole jméno) apod. [6]

2.5.5 Omezení

S využitím jistých pravidel, která jsou nad příslušnými sloupci, atributy, tabulkami či vztahy implementována implementujeme omezení. Ta mají za úkol zamezit přípustné datové hodnoty. Omezení jsou několika typů:

1) **Omezení primárního klíče –** omezení zaručující pomocí indexu jednoznačnou identifikaci záznamu.
2) **Referenční omezení –** zajišťuje přesnost nad vztahy mezi tabulkami. Je založeno na podstatě vynucení relace mezi tabulkami. Funguje na principu automatické kontroly, zda je každý primární klíč propojen s cizím klíčem, dále kontroly vkládání záznamu cizího klíče bez existence primárního klíče či zajištění odstranění záznamů v jedné tabulce. Tyto události jsou součástí upravení relací.
3) **Omezení integrity –** je to omezení zpřesňující data v databázi (viz. *Integrita dat v databázi*). Tato omezení jsou trojího typu:
    - **Omezení typu NOT NULL –** omezení, které zajišťuje, že hodnotu je nutno zadat.
    - **Omezení typu CHECK –** omezení, zajišťující platnost hodnoty, kde výsledkem je hodnota true (pravda) nebo false (nepravda).
    - **Omezení zajištěná pomoci spuštění –** slouží k vyhození určitého statusu při konkrétní události v databázi (například při vkládání záznamů). [5]

2.5.6 Datové typy

Datový typ je formátem daného pole (sloupce), který omezuje znaky ve sloupci na ty, které mají pro sloupec význam, zaručují určité chování sloupce a napomáhají k efektivnímu ukládání dat. [5] Tabulka je doplněna o základní datové typy programovacího jazyk Visual Basic for Application (VBA).

Datový typ	MS Access	MS SQL Server	Oracle	VBA
Znakový s pevnou délkou	TEXT	CHAR	CHAR	STRING
Znakový s proměnnou délkou	MEMO	VARCHAR	VARCHAR	STRING*100
Dlouhý textový	MEMO	TEXT	LONG	

Celočíselný	INTEGER (celočíselný) nebo LONG INTEGER (dlouhý celočíselný)	INTEGER nebo SMALLINT nebo TINYINT	NUMBER	INTEGER
Desítkový číselný	NUMBER (desetinné číslo)	DECIMAL nebo NUMBER	NUMBER	SINGLE nebo DOUBLE
Měna	CURRENCY	MONEY nebo SMALLMONEY	NUMBER	CURRENCY
Datum/čas	DATE/TIME	DATETIME nebo SMALLDATETIME	DATE nebo TIMESTAMP	DATE

Tabulka 2.1**: Datové typy v relačních databázích [5]**

2.6 Metodologie návrhu databáze - konceptuální návrh

Cílem metodologie návrhu databáze je, s využitím techniky a různých procedur nebo konkrétních nástrojů, usnadnění a zefektivnění návrhu databáze. Je postavena na třech základních fázích návrhu, a to na konceptuálním, logickém a fyzickém návrhu databáze. V této kapitole bude seznámeno s prvním z nich a to s konceptuálním.

Podstatou tohoto návrhu je základní definování entit, atributů nebo vztahů, a to bez ohledu na následnou implementaci do datových modelů nebo jinou fyzickou implementaci. Tento návrh je předáván logickému návrhu. [1]

2.6.1 Entita

Je to segment reálného světa, s nímž je zájem pracovat a je interesantní na takové úrovni, že jsou o něm shromažďována data, která jsou následně zapracována do databáze. Představivost u entity je prakticky neomezená, jedná se tak o cokoliv, o čem je možnost data shromáždit. Vyskytují se nejčastěji ve formě podstatného jména (příkladem entity je řidič). [5]

2.6.2 Omezení entity

Určují podmínky, které jednotlivé entity musí splňovat. (například musí být vyplněno jméno a příjmení řidiče). [10]

2.6.3 Atribut

Je to dále nedělitelný fakt, nesoucí příslušnou vlastnost entity, která je v rámci této entity uchovávána (příkladem atributu je jméno řidiče). Každá entita má povinnost obsahovat jedinečný atribut či skupinu atributů (též definovaných jako primární klíč). [5]

2.6.4 Doména

Neprázdná množina hodnot, která je specifikací atributu a nese tři základní vlastnosti:

1) **Obecné** – základní informace o poli, jako je například jméno a popis pole.
2) **Fyzické** – „viditelná“ podoba pole v databázi – například délka a datový typ pole.
3) **Logické** – popis hodnot uložených v poli – například rozsah hodnot. [9] [2]

2.6.5 Vztahy

Vztahy jsou dalším stěžejním pojmem ve světě databází. Pomocí vztahů je zajištěna propojenost jednotlivých tabulek. K tomuto propojení jsou využívány množiny primárních a cizích klíčů, případně i pomocí třetí tzv. vazební tabulky. [2]

U vztahů existují tři základní vlastnosti, a to kardinalita vztahu, volitelnost vztahu a stupeň vztahu. [10]

První z vlastností je kardinalita vztahu:

1) **Vztah jedna ku jedné (1:1)** – využití tohoto vztahu je při situaci, kdy záznam v tabulce jedna, lze propojit právě s jedním záznamem tabulky dva. Tento vztah je využíván jen zřídka, spíše svědčí o chybném návrhu databáze, protože obsahuje podobná data ve dvou různých tabulkách. Využití tohoto vztahu je především při doplňujících informacích u záznamů s rozdílnými situacemi (například tuzemští a zahraniční zaměstnanci – vytvoření zvláštní tabulky pro zahraniční zaměstnance).
2) **Vztah jedna ku více (1:N)** - nejvyužívanějším typem vztahu je právě 1:N. Tento vztah je využíván při situaci, kdy záznamu tabulky jedna je přiřazeno vícero záznamů tabulky dva a vícero záznamům tabulky dva, lze přiřadit jen jeden záznam tabulky jedna (jako příklad lze uvést databázi filmů, kde režisér může natočit několik filmů, ale film má pouze jednoho režiséra).
3) **Vztah více k více (M:N)** – speciálním vztahem je více k více. Speciálním je z toho hlediska, že v databázích nelze vytvořit přímou vazbu k tomuto vztahu. Je vytvářen v situacích, kdy vícero záznamům tabulky jedna, lze přiřadit vícero záznamů tabulky dva (příkladem této situace může být vztah mezi spisovateli a knihami, kde více knih napsalo více spisovatelů a více spisovatelů více knih). [6]

4) **Rekurzivní vztah** – je speciálním tzv. unárním vztahem[2] a je jen zřídka využívaný. Jedná se o vztah v rámci jedné tabulky a to v kombinaci všech tří předchozích vztahů, čili 1:1 (například zaměstnanec je manželem (manželkou) jiného zaměstnance), 1:N (například vztah nadřízených a podřízených zaměstnanců) a M:N (například sledování součástek při finálním výrobku). [5] [10]

Druhou vlastností je tzv. volitelnost vztahu. Ta představuje, zda tabulka je či není povinna obsahovat záznam související s druhou tabulkou.

Existují dva druhy účastí:

1) **Povinná** – představuje splnění povinnosti, že před vytvořením záznamů ve druhé tabulce musí být vytvořen záznam v tabulce první. Tím je ve vztahu zaručena povinná účast první tabulky.
2) **Volitelná** – představuje situaci, kdy k vytvoření záznamu v druhé tabulce není potřeba vytvářet záznam v první tabulce. Tím je ve vztahu zaručena volitelná účast první tabulky.

S účastmi ve vztahu jsou spjaty i tzv. stupně účastí. Stupeň účasti je určení nejvyššího a nejnižšího možného množství záznamů jedné tabulky v rámci vztahu s jedním záznamem tabulky druhé. Stupně účasti jsou nastavovány pomocí čísel. (například jeden záznam v první tabulce musí být ve vztahu minimálně k jednomu a maximálně k deseti záznamům v druhé tabulce je stupeň účasti zaznačen 1,10). [2]

2.7 Metodologie návrhu – logický návrh

Následujícím je logický návrh. V této fázi je konceptuální návrh zapracováván do tzv. datového modelu. V této práci je využíván relační databázový model, takže koncept je převeden do tabulek a relací s nimi spjatými.

Je založen na podstatě tzv. mapování entitně-relačních tabulek, kdy hlavním úkolem je na základě konceptuálního návrhu popsat jednotlivé tabulky, zkontrolovat je na základě normalizačních pravidel (viz kapitola 2.11), kontrola integrity dat (viz kapitola 2.10) a je zde potřebná kontrola podpory tzv. transakcí. [1]

[2] Unární vztah – vztah v rámci jedné tabulky

2.7.1 Transakce a její vlastnosti

Definice transakce zní: *„Transakce je diskrétní posloupnost kroků (operací), které musí být buďto všechny provedeny správně, nebo nesmí být provedeny vůbec.“* [5]

Transakce by měla vycházet z konzistentního stavu databáze a po dokončení by stav měl zůstat opět konzistentní. Dokončení transakce má dva stavy, a to pokud byla transakce úspěšná, tak se bude databáze nacházet v novém stavu (konzistentním). Pokud však transakce úspěšná nebude, tak bude transakce zrušena a databáze musí být v konzistentním stavu před jejím zahájením. [1]

Transakce mají čtyři základní vlastnosti pojmenovány zkratkou ACID:

1) **Atomicity (atomárnost)** – *„Transakce představuje dále nedělitelnou jednotku, která je buď úplně vykonána, nebo není vykonána vůbec“*. [1]
2) **Consistency (konzistence)** – tato vlastnost zaručuje konzistentní stav databáze jak před, tak i po transakci. Nezáleží zde však jen na konkrétní SŘBD, ale například i na lidském faktoru (například chybně zadaný účet pro bankovní převod ze strany programátora).
3) **Isolation (izolace)** – izolace je vlastností, která zajistí nezávislost transakcí v tom smyslu, že různé druhy a stavy transakcí neomezují jiné transakce.
4) **Durability (trvalost)** – zajistí trvanlivost záznamů po transakcích takovým způsobem, že záznamy nebudou z databáze ztraceny při případné havárii. [1]

2.8 Metodologie návrhu – fyzický návrh

Fyzický návrh je už konkrétní implementací logického návrhu do prostředí relačního SŘBD (v tomto případě MS Access), kde se využívá popis tabulek, indexace pro efektivnější vyhledávání, zajištění maximální integrity dat, implementaci uživatelských pohledů a požadavků apod. [1]

2.9 Integrita dat v databázi

Předcházení nepřesným, duplicitním, porušeným či chybným datům je hlavní vizí integrity dat. Patří k nejzákladnějším aspektům databáze, protože právě data jsou to, co dělá databázi silnější, a na základě těchto dat jsou učiněna důležitá rozhodnutí. Lze si snadno představit, že nejhorší možnou situací je učinění rozhodnutí dle chybných dat.

V procesu návrhu jsou implementovány čtyři hlavní integrity:

1) **Entitní integrita** – opatřuje neexistenci duplicitních záznamů, dále pak určení jedinečnosti polí záznamů a zajištění nenulových v rámci těchto polí.
2) **Doménová integrita –** pomocí této integrity je zajištěna přesnost, spolehlivost, platnost a konzistentnost daného pole.
3) **Referenční integrita –** má na starost opatření synchronizovaného vztahu mezi tabulkami, především při zadávání, mazání a úpravě dat v jednotlivých tabulkách. [2]
4) **Business pravidla –** souvisejí s chodem dané organizace a způsobem získávání dat do databáze. Na základě způsobu získávání dat jsou zaváděny různá omezení, která ovlivňují určité fáze návrhu databáze.[5]

2.10 Normalizace

Principiálně se jedná o rozložení tabulek za účelem zvýšení přehlednosti, rychlosti vyhledávání, ale se zachováním původního schématu a jejich vztahů. Na základě těchto aspektů, je cílem zabránění opakujících se dat, chybám při aktualizaci záznamů apod. [10]

2.10.1 První normální forma (1NF)

Definice první normální formy vychází z několika aspektů, které musí tabulka splňovat. Pokud je splněno, že v tabulce se nevyskytují žádné shodné záznamy, jsou nadefinována klíčová pole, všechny atributy jsou závislé na primárním klíči a je splněna podmínka nedělitelnosti sloupců, tak je tabulka v první normální formě. [10]

2.10.2 Druhá normální forma (2NF)

„Tabulka je v 2NF, když je v 1NF a každý neklíčový atribut je plně závislý na primárním klíči, a to na celém klíči a nejen na nějaké jeho podmnožině. Aby to bylo možné, musí se primární klíč skládat z více než jednoho atributu. Pokud tedy primární klíč zahrnuje jen jedno pole, tabulka je automaticky ve 2NF, pokud splňuje 1NF.“ [10]

2.10.3 Třetí normální forma (3NF)

Další formou je v pořadí třetí. Vychází opět z předcházející definice, tedy druhé normální formy. Je to forma, která nad rámec druhé normální formy nesmí obsahovat tzv. tranzitivní závislosti. *„Za tranzitivně závislý považujeme takový atribut, který je závislý na jiném atributu, jenž sám není primárním klíčem “své“ relace“*. Jinými slovy zde existuje nezávislost mezi neklíčovými atributy. [5]

2.10.4 Boyce Coddova normální forma (BCNF)

Jako u všech předchozích, tak i zde je základem této formy forma předcházející, tedy tabulka se nachází v Boyce Coddově normální formě právě tehdy, je-li i ve třetí normální formě, ovšem zpětně tomu tak není. Jestliže třetí forma specifikovala neúčast tranzitivních závislostí, tato forma je formulován tak, že spolu s tranzitivními nesmí obsahovat taky závislosti mezi kandidátními klíči.

Za porušení této normální formy se pokládá více kandidátních klíčů v tabulce, kde minimálně dva klíče musí být složeny z více atributů a zároveň některé kandidátní klíče, které jsou složené, musí mít společný atribut. [10]

2.10.5 Čtvrtá normální forma (4NF) a pátá normální forma (5NF)

Tyto normální formy, jsou v praktickém životě málo využívány, a tak postačí, že čtvrtá normální forma se zakládá na nespojitosti nezávislých skupin v jedné tabulce a pátá normální forma je založena na pravidlu, kde je zakázáno přidat další atributy takovým způsobem, který by zapříčinil rozložení tabulek na dílčí vlivem nepředpokládaných událostí. [10]

2.11 Faktory efektivního návrhu a chodu databáze

2.11.1 Konceptuální přehled při návrhu databáze

Správnými postupy a dodržením metodologie návrhu bude databáze přehledná, konzistentní a bude zajišťovat vysokou integritu dat. K této skutečnosti napomáhají základní faktory, kde jejich splnění napomáhá k zajištění efektivnímu návrhu a tedy i chodu databáze. Mezi tyto faktory patří:

- **Formulace úkolů a cílů** – definování k čemu má databáze sloužit (formulace úkolů) a jaké úkony budou uživatelem databáze prováděny (formulace cílů).
- **Analýza existující databáze** – pokud databáze již dříve existovala (ať už v papírové nebo elektronické podobě), tak poskytuje informace o chodu organizace a využití dat v ní, což napomůže k vylepšení právě vytvářené databáze. Další fází analýzy jsou rozhovory s uživateli a managementem organizace o fungování stávající a návrhu a zlepšení budoucí databáze.
- **Vytváření datových struktur –** zde se jedná o pečlivé definování jednotlivých tabulek, polí uvnitř tabulek podle přesných požadavků a pravidel a určení

jejich omezení či specifikací. I zde je na místě konzultace s managementem, především o vlastnostech jednotlivých polí.

- **Zajištění vztahů mezi tabulkami –** po opětovné konzultaci jsou mezi tabulkami vytvořeny relace, jejich vlastnosti a využití integrity vyhovující požadavkům organizace.
- **Definování business pravidel –** zjištění faktů a omezení specifických pro chod organizace. Jedná se o specifikaci a určení podmínek pro jednotlivé záznamy, či pole po konzultaci s uživateli či managementem.
- **Definování pohledů** – je „překvapivě" výsledkem diskuzí s managementem a uživateli databáze v organizaci a specifikuje konkrétní požadavky na pohledy. Například jeden uživatel pracuje s komplexnějším pohledem než jiný.
- **Kontrola integrity dat** – následným krokem je celková a důkladná kontrola integrity dat a to jak na úrovni jednotlivých tabulek, tak polí či vztahů (viz podkapitola 2.10 Integrita dat v databázi). [2]
- **Proces normalizace** – v neposlední řadě, ale ze základních pravidel jedna z nejpodstatnějších operací je tzv. normalizace (viz podkapitola 2.11 Normalizace databází). [1]

2.11.2 Čeho se při návrhu vyvarovat

Tak jako v životě, tak i ve světě návrhu databází jsou určité aspekty, kterým je potřeba se vyvarovat. Těchto aspektů je mnoho, zde jsou nastíněny tři nejzákladnější:

1) **Styl nestrukturovaného souboru** – je styl návrhu, kde jsou všechna data umístěna do jedné velké tabulky. S touto možností se skýtá řada problémů, kde například neexistuje unikátnost záznamů, jsou zde vícesložková pole (například jméno zákazníka obsahuje křestní jméno i příjmení), mohou se zde vyskytovat pole s vypočítanou hodnotou (například velikost jedné objednávky) aj. Na první pohled je zde vidět neprovázanost mezi záznamy, složitý vyhledávací proces a možnost tvorby duplicitních dat.
2) **Návrh ve stylu tabulkového procesoru** – mnoho návrhů ztroskotá na převzetí dat z tabulkového procesoru. Je potřeba se tomuto návrhu vyvarovat a data pečlivě „rozdělit" do jednotlivých tabulek provázaných relacemi. Při nerespektování a převzetí tabulkového návrhu zde hrozí zdlouhavost

vyhledávacích procesů, duplicita či nekonzistentnost dat a tím pádem neefektivnost návrhu databáze.

3) **Návrh založený na databázovém softwaru** – je preferován, především u uživatelů, kteří mají s konkrétní aplikací (například MS Access) zkušenosti a řídí se při návrhu jejími jednotlivými vlastnostmi či nástroji. Při navrhování databáze by měl být návrh nezávislý na konkrétním SŘBD, proto je dobré se tomuto postupu vyvarovat. Mohou tak vzniknout problémy typu řízení návrhu pomocí SŘBD a únik od požadavků organizace, daný SŘBD nemusí být vhodný pro daný typ organizace, provádění rozhodnutí na základě dovednosti daného systému aj. [2]

2.11.3 Výhody při správném návrhu databáze

Kompletní využití a znalost metodologie návrhu staví celkovou strukturu databáze, její vlastnosti a celkovou funkčnost do popředí a zvyšuje tak její důvěryhodnost a efektivitu. Mezi výhody spojené se správným využitím metodologie patří například:

- snadný přístup k informacím na základě jednoznačných dotazů a stanovené struktury,
- data jsou integrovaná, bez duplicitních či tranzitivních hodnot,
- úpravy, aktualizace, či mazání jednotlivých záznamů mohou probíhat současně v několika tabulkách apod. [2]

2.12 Microsoft Access

Microsoft Access je aplikace pro efektivní ukládání dat a využívání informací, jako nástupce kartotékového či pořadačového zpracování. Nástroje v této aplikaci jsou koncipovány pro efektivní, přehledné a hlavně rychlé vyhledávání informací na základě relačního databázového modelu.

MS Access patří do balíčku programové rodiny MS Office, kde jeho blízkými jsou programy MS Word, MS Excel či MS PowerPoint, kde každou tuto aplikaci lze využít pro různé zpracování informací.

Nejvíce společného má Access právě s tabulkovým procesorem Excel. Naskýtá se zde otázka, jestli není Excel dostačující, a jaký je důvod vytváření a učení se pojmů světa databází. Odpověď je více než jednoduchá, a to, že databáze dokáže data lépe evidovat a navíc efektivněji využívat tyto data a to i při jejich velmi rozsáhlém počtu. [8]

Při vytváření databáze v Accessu bude využíváno objektů databáze. Mezi tyto objekty

patří tabulky, formuláře, sestavy či dotazy. Jednotlivé objekty budou představeny v jednotlivých kapitolách, nicméně mají společný způsob zobrazování a vytváření:

V Accessu existuje několik forem zobrazení a to:

1) **Návrhové zobrazení –** v tomto zobrazení lze objekty vytvářet, modifikovat, přidávat nové prvky, či definovat jejich jednotlivé vlastnosti (například u objektu Tabulka definovat velikost pole, vstupní masku[3] apod.).
2) **Zobrazení datového listu** – zobrazení přejímající vlastnosti polí nadefinovaných v návrhovém zobrazení a do těchto polí jsou již vkládána příslušná data (například v objektu tabulka do pole „rok“ je vložena hodnota „2014“).
3) **Zobrazení rozložení –** je kompromise mezi návrhovým zobrazením a zobrazením datového listu, kdy už je zřetelná finální podoba, ale je zde stále možnost optimalizací a nelze přidávat nové záznamy (rozložení se využívá převážně u formulářů).
4) **Zobrazení kontingenční tabulky –** je tabulkou sloužící ne k uchovávání dat, ale k jejich analýze. Pomocí ní lze v Accessu například zjistit, kolik zaměstnanců v databázi má vysokoškolské vzdělání apod.
5) **Zobrazení kontingenčního grafu** – jedná se o grafické zobrazení kontingenční tabulky. [6]

Se zobrazením je úzce spjato i vytváření jednotlivých objektů. V tomto směru nabízí Access také několik možností:

1) **Automatické vytvoření objektu –** pomocí daného příkazu je objekt automaticky vytvořen bez větších nároku na estetiku.
2) **Vytváření pomocí průvodce –** vytvořením daného objektu provází krok za krokem prostředí Accessu, které nabízí v jednotlivých krocích jednotlivé funkce či vlastnosti, které bude daný objekt obsahovat.
3) **Objekt vytvářený ručně v návrhovém zobrazení –** zde je konkrétní objekt nadefinován a vytvořen podle vlastní požadavků. Od nastavení velikosti, rozmístění prvků objektu až po přidání vlastností jednotlivým prvkům [6]

[3] Vstupní maska – přesně definuje formát daného pole (například čtyři místa pro zadání daného roku)

2.11.1 Tabulky

S tabulkami již bylo seznámeno dříve, a tak spíše než s definicí tabulek se tato podkapitola bude zabývat jejími vlastnostmi v Accessu.

Co však nebylo úplně objasněno, je fakt, že každá tabulka by měla mít nejen specifické jméno v rámci databáze, ale také jméno naprosto a přesně vystihující danou problematiku tabulky, a to nejen z důvodu přehlednosti a efektivnosti, ale i pro zajištění přesné práce s danou databází. [4]

Vytváření tabulek nabízí na rozdíl od jiných objektů ještě tyto způsoby vytvoření:

1) **V zobrazení datového listu –** vytvořit tabulku lze možná ještě snadněji, a to pomocí příkazu „Vytvoření“ přímo v datovém listu a lze tak přímo vidět jak se vytvářejí jednotlivé sloupce a lze jim intuitivně přidávat i vlastnosti.
2) **Zadáním dat v zobrazení datového listu –** podobný styl jako předchozí způsob „V zobrazení datového listu“ ovšem s podstatným rozdílem, že jsou zadávána konkrétní data tabulky a Access rozpozná vlastnost konkrétního pole. [6]

Mezi základní vlastnosti tabulky v Accessu patří datové typy. Datové typy již byly definovány v podpodkapitole 2.6.6, ale každý datový typ má možnost nastavení konkrétních vlastností:

- **velikost pole** – přiděluje konkrétní místo v poli. U textu to jsou znaky, u čísel je to přesnost čísla (například dlouhé celé číslo),
- **formát** – určuje, jak bude údaj v poli zobrazen (například zobrazení data ve formátu měsíc/rok),
- **vstupní maska –** rozdíl oproti formátu je v tom, že jednoznačně určuje, které hodnoty lze do pole zadat a jak budou uloženy v tabulce,
- **titulek –** slouží k popisu pole,
- **výchozí hodnota –** hodnota, které bude automaticky nastavena v daném poli,
- **ověřovací pravidlo –** určuje kritéria, které musí údaj splňovat (například pole věk musí být kladné),
- **ověřovací text –** text, který se objeví ve formě chybového hlášení po nesplnění ověřovacího pravidla,

- **je nutno zadat –** nabývá dvou stavů (ANO, NE), které zajišťují, zda je povinnost údaj v poli zadat či ne,
- **povolit nulovou délku –** povoluje či zakazuje zadat text s nulovou délkou. Pokud nulová délka není definována jako nevyplněné pole, ale pomocí uvozovek, tak informuje o prázdném poli, tedy o jeho úmyslném nevyplnění,
- **indexovat –** umožňuje rychlejší vyhledávání záznamů, a to dle třech stavů:
  - **a.** ne – nastavení pokud pole není potřeba indexovat,
  - **b.** ano (duplicita povolena) – pokud je potřeba vyhledávání záznamů s duplicitními poli (například indexování pole příjmení, která jsou stejná),
  - **c.** ano (bez duplicity) - pokud je vyžadováno jednoznačné řazení či vyhledávání záznamů (například indexování pole rodné číslo),
- **komprese kódu UNICODE –** komprese textů z velikosti dvou bajtů na jeden, kde délka jednoho bajtu je typická pro většinu jazyků,
- **režim editoru IME –** režim, který se týká podpory asijských verzí,
- **režim sentence IME –** určuje typ dat při vkládání hodnot v režimu IME,
- **inteligentní značky –** možnost podporující funkce (například zadání parcely v hektarech a pomocí značky ji lze přepočítat na metry čtvereční). [4]

Dalšími vlastnostmi jsou operace s tabulkami, jako je přidání sloupce, přesunutí sloupce, skrytí sloupce, které jsou však naprosto intuitivní. Pokud bude potřeba změnit struktura tabulky, tak lze jednoduše a intuitivně v „Návrhovém zobrazení“. [6]

2.11.2 Formuláře

Dalším z objektů Accessu jsou formuláře. Přejímají data z tabulky (či více tabulek) a dotazů, za účelem větší přehlednosti a minimalizace vytváření chyb z důvodu nepozornosti. Je zde zřetelná viditelnost zapisování dat do konkrétního, přesně popsaného a uspořádaného pole. [6]

Je potřeba také zmínit, že formuláře neuchovávají data jako taková, ale přejímají je z daných tabulek či dotazů. Formuláře jsou také vybaveny různými ovládacími prvky, které pomáhají k zobrazování určitých dat. Tyto prvky lze ve formulářích libovolně umísťovat. [10]

Prvky jsou dvojího typu:

1) **Vázané** – prvky, které zobrazují údaje mající vztah k datům v databázi (například pole jméno).

2) **Nevázané** – prvky, které zobrazují data nemající žádný vztah k databázi (příkladem jsou různé přepínače). [6]

Tak jako pole v tabulkách, tak i prvky ve formulářích mají určité vlastnosti. Tyto vlastnosti jsou rozděleny do čtyř sekcí, ve kterých lze tyto vlastnosti nastavovat:

1) **Formátové –** vlastnosti související s formátem prvků (například nastavení viditelnosti prvku či nastavení zobrazení prvku).
2) **Událostní –** v této sekci jsou nastavovány jednotlivé události související s prvky (například nastavení prvku „tlačítko“ na formuláři tak, že po jeho stisknutí se otevře konkrétní sestava).
3) **Datové –** zde je možnost nastavení vlastností souvisejících s daty (například povolení či zakázání vymazání dat).
4) **Jiné –** zde jsou nastavovány doplňující vlastnosti (například kam má „přeskočit“ kurzor v posledním poli formuláře po stisknutí klávesy TAB). [10]

2.11.3 Sestavy

Data již byly zapracovány do databáze, a nyní se skýtá otázka jak tyto data publikovat. Publikace však neznamená jen získání dat na monitoru, ale také například na papír. K tomuto účelu se využívá další objekt a to sestava. Lze je využít například na tisk objednávek, faktur apod. [6]

I sestavy mají určité vlastnosti, které lze stejně jako ve formulářích nastavovat pomocí definovaných čtyř sekcí, ale mají i své samostatné vlastnosti jako jsou například:

- řazení záznamů zde neřídí zdroj záznamů jako ve formuláři, ale lze záznamy v sestavě libovolně seřadit,
- sestavy jsou určeny jen pro tisk, takže v nich nelze data měnit,
- v rámci sestav je možno definovat několik záhlaví a zápatí na jedné stránce sestavy, což umožňuje například rozdělení studentů do jednotlivých předmětů. [10]

2.11.4 Výběr dat

Databáze slouží nejen jako úložiště dat, ale je potřeba z nich i získávat informace. Pro získání informací slouží dva způsoby: [6]

1) **Výběr dat pomocí filtru** – slouží k jednoduchému a rychlému vyhledání dat z určité skupiny dat pomocí určovacích kritérií (například výběr zákazníku z určitého města). Filtrování je rozděleno do několika druhů:
    - ***filtr podle výběru*** – v poli, dle kterého bude filtrováno, se označí kritérium filtrace a pomocí příkazu se vyfiltrují dané záznamy,
    - ***filtr mimo výběr*** – je podobný předchozímu filtrování, ale pole se po výběru nezobrazí, ale skryje. Jsou tedy vybrány všechny pole kromě vybraného,
    - ***filtr podle formuláře*** – výběr dat pomocí formuláře, kde jsou hodnoty vybírány ze seznamu. Výhodou je zde kombinace jednotlivých kritérií (například název města i ulice),
    - ***rozšířený filtr*** – získávání dat podle složitějších kritérií, které už se blíží dotazům. [6]
2) **Výběr dat pomocí dotazů –** získávání dat pomocí dotazů je v podstatě podobné s filtry, nicméně je zde řada rozdílů. Zatímco filtry se ukládají jako vlastnosti tabulky, tak dotazy jsou samostatnými objekty, které lze znovu využívat. Dalšími rozdíly jsou například možnost využití dotazů k úpravě v databázi, kde lze výsledku dotazu využít jako zdroje dalšího dotazu či dotazy mohou vybírat data z celé databáze a zobrazovat jen vybraná pole. [6]

    Druhy dotazů jsou tyto:

    - ***výběrový dotaz*** – výsledkem dotazu jsou data z tabulky, popřípadě tabulek, [6]
    - ***křížový dotaz*** – výsledkem dotazu je dvourozměrná tabulka. Je zde podobnost s kontingenční tabulkou a slouží tedy k jakési analýze dat (například výpočet průměrné známky studenta), [10]
    - ***akční dotazy*** – slouží k úpravě dat v databázi a v Accessu jsou čtyři:
        - *odstraňovací dotaz* – po výběru dle kritérií odstraní záznamy,
        - *aktualizační dotaz* – možnost úpravy dat dle kritérií,
        - *přidávací dotaz* – možnost přidání záznamu dle kritérií,
        - *vytvářecí dotaz* – vytvoření nového záznamu (tabulky), [6]

- ***dotazy jazyka SQL*** – je to univerzální způsob dotazování, pomocí zadávání jednotlivých příkazů jazyka SQL (například příkaz SELECT). [4]

2.11.5 Makra

Makro je definováno jako určitý sled událostí, které jdou za sebou a jdou snadno vytvářet i z důvodu vytváření příkazů pomocí češtiny (česká verze Accessu). [3]

Makra však skýtají řadu omezení, a ne vše lze pomocí maker vytvořit. Proto lze makra převést do jazyka VBA, který již zajistí splnění požadavků. Převedení však nelze učinit v opačném směru, tedy převedení z jazyka VBA do prostředí maker. [4]

2.11.6 Programové moduly a jazyk VBA

Definice programových modulů se opírá o jazyk VBA. Jsou to tedy procedury a funkce, které jsou psány v rámci programového jazyka VBA (Visual Basic for Application). Tyto modulu již obsahují větší možnosti především díky neomezené práci databází. [3]

Moduly jsou rozdělovány na dva typy:

1) **Moduly třídy –** moduly využívány u formulářů či sestav, výjimečně samostatně. Jsou psány v rámci vlastností jednotlivých prvků, formulářů apod. či událostních procedur.
2) **Moduly standardní** – jsou využívány pro univerzální funkce či procedury. [10]

Jazyk VBA je, na rozdíl od předešlých příbuzných jazyků, jako byl například jazyk VBE (Visual Basic Editor), mnohem intuitivnější, strukturovanější, robustnější, objektově orientovanější a lze s jeho pomocí vytvořit i příkazy, které standardně nelze v Accessu vytvářet. [7]

3 Analýza současného stavu

Kapitola seznamující s aktuální stavem vedení dat v klubu a představením klubu jako takového, který vznikl v roce 1926 jako vyústění zdejších složek sokola a orla.

3.1 Představení Handball Clubu Zubří

Handball Club Zubří je házenkářský klub, kde pozornost je zaměřena pouze na provozování mužské složky, a to od nejmenších „předškoláků“ až po extraligové „A mužstvo“.

Provoz klubu HC Zubří je rozdělen do čtyř subjektů, a to na Valašský házenkářský klub a.s., který zajišťuje chod „A družstva“ mužů, dále pak HC Zubří o.s. zajišťující provoz všech mládežnických kategorií až na dorostenecké, které patří pod projekt SCM (sportovní centrum mládež), který přispívá a financuje chod dorosteneckých družstev z účelových prostředků MŠMT. Poslední složkou je nadační fond Talent házená zřízený pro podporu a chod mládežnických mužstev.

Družstvo můžu je účastníkem nejvyšší tuzemské soutěže v házené, kde dlouhodobě patří k nejlepším týmům, o čemž dokazuje zisk tří titulů mistrů republiky v letech 1996, 1997 a 2012 a zisk několika stříbrných medailí. Nejvyšší tuzemské soutěže hrají i mládežnické kategorie, kde patří rovněž k nejlepším v republice a pravidelně se umísťují na předních příčkách. Důkazem může být například zisk mistrovských titulů v kategoriích mladší a starší dorost v roce 2007, zisku titulu mistra ČR v kategorii starší dorost v roce 2008 či zisk titulu mistra ČR v kategorii starší žáci v roce 2004.

3.2 Analýza současného stavu

V rámci klubu je momentálně registrováno celkem šest kategorií, a to od těch nejmenších, tedy kategorie přípravky, přes mini-žáky, mladší žáky, starší žáky, mladší dorostence, starší dorostence až po nejvyšší kategorii mužů. Celkem je v klubu momentálně registrováno 149 hráčů, 9 licencovaných trenérů a 11 funkcionářů.

O všech hráčích jsou vedeny záznamy pouze kartotékovým způsobem, maximálně s využití tabulkového procesoru, a to především z důvodu jakéhosi zvyku. Tento způsob bývával populární, ale v dnešním světě informačních technologií, který naskýtá nespočet možností a je neustále vyvíjen astronomickou rychlostí kupředu, existují i daleko rafinovanější způsoby zpracování dat. Jedním ze způsobu řešení těchto nedostatků, jako je nadbytečnost záznamů, možnost časté duplicity dat či komplikované upravování záznamů jako reakci na změny okolí, je relační databáze. Ta, na rozdíl od kartoték a tabulkových

procesorů, umožňuje mezi daty vytvářet vztahy a umožnit tak uživateli pohodlnější, rychlejší a také přesnější vyhledávání záznamů než tomu je u kombinace předchozích možností. Dále lze například snadno a efektivně upravit jednu či více hodnot a změny se okamžitě promítnou do všech záznamů spjatých s touto hodnotou.

Záznamy jsou vedeny od kategorie mladších žáků výše, tedy v rámci pěti kategorií. Ke každému hráči jsou vedeny osobní údaje, statistiky z jednotlivých zápasů, přehled o vypůjčených klubových potřebách (například klubové oblečení nebo házenkářský míč) apod.

4 Návrh a implementace databázové aplikace

Po získání teoretických znalostí a zhodnocení aktuálního stavu zpracování dat je na čase přejít k zásadní kapitole, která se zabývá návrhem a implementací konkrétní databázové aplikace.

Základním kamenem každé databáze jsou především data. Aplikace může být dokonale navržena, ale pokud obsahuje nepřesná, nepravá či poškozená data, tak její funkčnost je značně omezena. Z tohoto důvodu, je sběr správných dat prvním základním krokem k vytvoření úspěšné databáze. V této aplikaci bylo využito prvotních dat, kterých bylo získáváno přímo od jednotlivých hráčů v osobním kontaktu, a to prostřednictvím předem nadefinovaných formulářů. Dalšími zdroji byly zápisy ze zápasů, odpovídající přiřazení hráčům k jednotlivým zápasům a klubová kartotéka potřeb pro hráče.

Jelikož v klubu existovaly záznamy o hráčích, ale jejich kvalita byla dosti pofidérní a především neúplná, bylo potřeba tyto data získat. Šlo především o data jako je výška a váha každého z hráčů, kontaktní údaje, které někteří hráči, v rámci klubové kartotéky neměli vyplněné, a to ani údaje o svém bydlišti. Proto, jak již bylo řečeno, bylo nutností informace získat z prvotních zdrojů, aby nedocházelo k ukládání nepřesných dat.

4.1 Návrh tabulek

Přesná, pravdivá a úplná data jsou základním kamenem každé databáze, nicméně je potřeba určitým způsobem tyto data ukládat. Pro ukládání dat bylo vytvořeno celkově devět tabulek. Databáze je založena na dvou základních tabulkách *tblHraci* a *tblZapasy* se kterými jsou spjaty tabulky ostatní, tedy *tblZapasHrac*, *tblStatistikaHraci*, *tblStatistikaBrankari*, *tblVeci*, *tblTypyVeci*, *tblKategorieHrace* a *tblPostHrace*.

4.1.1 Návrh tabulky tblHraci

Tabulka obsahuje osobní informace o jednotlivých hráčích, ze kterých je čerpáno jako základ pro jiné tabulky. Tabulka obsahuje šestnáct atributů, kde každý záznamy by měl splňovat jedinečnosti na základě primárního klíče. Tuto jedinečnost zajišťuje atribut *RodneCisloHrace*, který je zadáván dle vstupní masky. Jelikož je označen jako primární klíč, bylo potřeba zadat vlastnosti nutnosti vyplnění, zakázat vlastnost povolení nulové délky a zatrhnout vlastnost indexovat bez duplicitních záznamů. Dalšími textovými atributy jsou *PrijmeniHrace*, *JmenoHrace*, *UliceHrace*, *CisloPopisneHrace*, *MestoHrace*, *ZemeHrace*, *EmailHrace* a *PostHrace* u kterého je zdrojem řádků samostatná tabulka *tblPostHrace*. Posledním textovým datovým typem, v rámci tabulky, je *KategorieHrace*, kde zdrojem řádků

je rovněž samostatná tabulka, konkrétně *tblKategorieHrace.* Všechny zmíněné atributy splňují podmínku nutnosti zadání až na atributy e-mailu a telefonního čísla hráče, a to z důvodu, že určití hráči nebyli ochotni poskytnout kontaktní údaje třetím osobám. Dalším atributem je *DatumNarozeni.* Pole splňuje i podmínku zákazu zadání budoucího data. Zbývající pole tabulky jsou číselné datové typy. *CisloHrace*, se vstupní maskou, která zajišťuje velikost čísla hráče do velikosti čísla sto, a taktéž i kladnou hodnotu čísla. *PscMestaHrace* opět má vstupní masku, pomocí níž je zajištěno přesné zapsání poštovního směrovacího čísla, *TelefonniCisloHrace* rovněž obsahuje masku pro korektní zadání čísla, *VyskaHrace* a *VahaHrace* je zadávána jak jinak než dle vstupní masky. Všechny číselné datové typy je také nutno zadat s výjimkou telefonního čísla hráče.

Název pole	**Datový typ**	**Vstupní maska**	**Nutno zadat**	**Indexovat**
RodneCisloHrace	Text	000000\/0009;;_	Ano	Ano (bez duplicity)
PrijmeniHrace	Text		Ano	Ne
JmenoHrace	Text		Ano	Ne
DatumNarozeni	Datum a čas		Ano	Ne
CisloHrace	Číslo	99	Ano	Ne
KategorieHrace	Text		Ano	Ne
PostHrace	Text		Ano	Ne
UliceHrace	Text		Ano	Ne
CisloPopisneHrace	Text		Ano	Ne
MestoHrace	Text		Ano	Ne
ZemeHrace	Text		Ano	Ne
EmailHrace	Text		Ne	Ne
TelefonniCisloHrace	Číslo	"+420 "999\ 999\ 999	Ne	Ne
VyskaHrace	Číslo	999" cm";;_	Ano	Ne
VahaHrace	Číslo	999" kg";;_	Ano	Ne

Tabulka 4.1: Vlastnosti tabulky tblHraci

4.1.2 Návrh tabulky tblZapasy

Druhá tabulka obsahuje informace o jednotlivých zápasech. Klíčovým polem, které je definováno jako primární klíč je *KodZapasu.* Je složen ze dvou částí, přičemž první část tvoří dvoumístná zkratka kategorie, ve které je zápas odehrán, a následující čtyři místa jsou rezervována pro kód konkrétního zápasu. Z důvodu správného zadání, byla vytvořena vstupní maska. Dalšími poli jsou *NazevZapasu*, *VysledekZapasu*, se vstupní maskou. Dalšími jsou *DatumZapasu* a *CasZapasu*. Všechny pole v tabulce je nutno zadat.

Název pole	**Datový typ**	**Vstupní maska**	**Je nutno zadat**	**Indexovat**
KodZapasu	Text	>LL0000	Ano	Ano (bez duplicity)
NazevZapasu	Text		Ano	Ne
VysledekZapasu	Číslo	00\:00;;_	Ano	Ne
DatumZapasu	Datum a čas		Ano	Ne
CasZapasu	Datum a čas		Ano	Ne

Tabulka 4.2: Vlastnosti tabulky tblZapasy

4.1.3 Návrh tabulky tblZapasHrac

Jelikož bude vytvářeno propojení tabulek a každý hráč může hrát ve více zápasech a každý zápas má více hráčů bylo potřebné vytvořit tzv. propojovací tabulku. Bude obsahovat pouze dvě pole, a to cizí klíče z tabulek *tblHraci* a *tblZapasy.* Aby bylo možné tabulku propojit, je nutno u vlastnosti polí každému poli nastavit vlastnost *Indexovat (duplicita povolena)*. Tato vlastnost zaručuje, že kombinace těchto dvou polí, tedy složený primární klíč, bude zajišťovat jedinečnost záznamu, ale každé jednotlivé pole z primárního klíč může být definováno vícekrát v rámci tabulky.

Název pole	**Datový typ**	**Vstupní maska**	**Je nutno zadat**	**Indexovat**
KodZapasu	Text	>LL0000	Ano	Ano (duplicita povolena)
RodneCisloHrace	Text	000000\/0009;;_	Ano	Ano (duplicita povolena)

Tabulka 4.3: Vlastnosti tabulky tblZapasHrac

4.1.4 Návrh tabulky tblVeci

Každý hráč dostává od klubu určité potřeby a věci. Přehled o těchto potřebách je zaznamenáván do tabulky *tblVeci*. Jako všechny, tak i tato tabulka obsahuje primární klíč. Jelikož je předpokládáno vytvoření vztahu s tabulkou hráčů, je primární klíč vytvořen kombinací dvou polí, a to cizím klíčem tabulky *tblHraci*, tedy polem *RodneCisloHrace* a polem *TypVeci.* U každého z polí primárního klíče je potřeba opět zatrhnout vlastnost *Indexovat (duplicita povolena)* aby byla zajištěna vazba na tabulku *tblHraci*. Pole *typVeci* je textovým datovým typem a bylo vytvořeno tak, že zdrojem řádku je samostatná tabulka *tblTypyVeci*. Dalšími atributy jsou *PocetVeci*, se vstupní maskou, *VelikostVeci*, *DatumVeci* kde je zadáván datum dostání určité věci. Opět je zde podmínka nutnosti zadání všech atributů v rámci záznamu.

Název pole	**Datový typ**	**Vstupní maska**	**Je nutno zadat**	**Indexovat**
TypVeci	Text		Ano	Ano (duplicita povolena)
RodneCisloHrace	Text	000000\/0009;;_	Ano	Ano (duplicita povolena)
PocetVeci	Číslo	0	Ano	Ne
VelikostVeci	Text		Ano	Ne
DatumVeci	Datum a čas		Ano	Ne

Tabulka 4.4: Vlastnosti tabulky tblVeci

4.1.5 Návrh tabulky tblStatistikaHraci

U každého hráče je potřeba zadávat jeho statistiky z jednotlivých záznamů. Pro tento účel byla vytvořena tabulka *tblStatistikaHraci.* Opět se zde opakuje situace se složeným primárním klíčem, a tedy i vlastnost duplicitního indexování klíče. Složen je z cizích klíčů tabulek *tblZapasy* a *tblHraci*, tedy polí *KodZapasu* a *RodneCisloHrace.* Dalšími atributy jsou číselné datové typy, u kterých je definovaná jednotná vstupní maska *0.* Do každého pole je zadáván počet daných akcí. Názvy těchto polí jsou *PocetStrel*, *PocetGolu*, *ZtrataMice*, *ZiskMice*, *PlusVobrane*, *MinusVobrane*, *Zisk7mHodu*, *Zavineni7mHodu*, *Zisk2minut*, *Zavineni2minut*, *ZiskZluteKarty*, *ZavineniZluteKarty*. Již opakovaná vlastnost nutnosti zadáni je i v této tabulce.

Vlastnosti popisuje následující tabulka.

Název pole	Datový typ	Vstupní maska	Je nutno zadat	Indexovat
KodZapasu	Text	>LL0000	Ano	Ano (duplicita povolena)
RodneCisloHrace	Text	000000\/0009;;_	Ano	Ano (duplicita povolena)
Ostatní pole tabulky	Číslo		Ano	Ne

Tabulka 4.5: Vlastnosti tabulky tblStatistikaHraci

4.1.6 Návrh tabulky tblStatistikaBrankari

Jelikož brankáři mají odlišné statistky od hráčů, tak je pro tento účel vytvořená samostatná tabulka. Vlastnosti primárního klíče jsou stejné jako u předchozí tabulky *tblStatistikaHraci.* Definice dalších polí se opět odvíjí od předchozí tabulky, a to i s datovými typy a vstupními maskami. Názvy těchto polí jsou *PocetStrel*, v souvislosti s počtem vystřelených střel soupeřem, *PocetGolu*, kde záznamy v tomto poli jsou ve spojitosti s inkasovanými góly, *PocetZakroku*, odpovídající počtu zákroků brankáře. Ostatní atributy *ZtrataMice* a *ZiskMice*, již jasně odpovídají názvu a jsou stejné jako u statistik hráčů. Zadání všech atributů je opět povinné.

Název pole	Datový typ	Vstupní maska	Je nutno zadat	Indexovat
KodZapasu	Text	>LL0000	Ano	Ano (duplicita povolena)
RodneCisloHrace	Text	000000\/0009;;_	Ano	Ano (duplicita povolena)
Ostatní pole tabulky	Číslo		Ano	Ne

Tabulka 4.6: Vlastnosti tabulky tblStatistikaBrankari

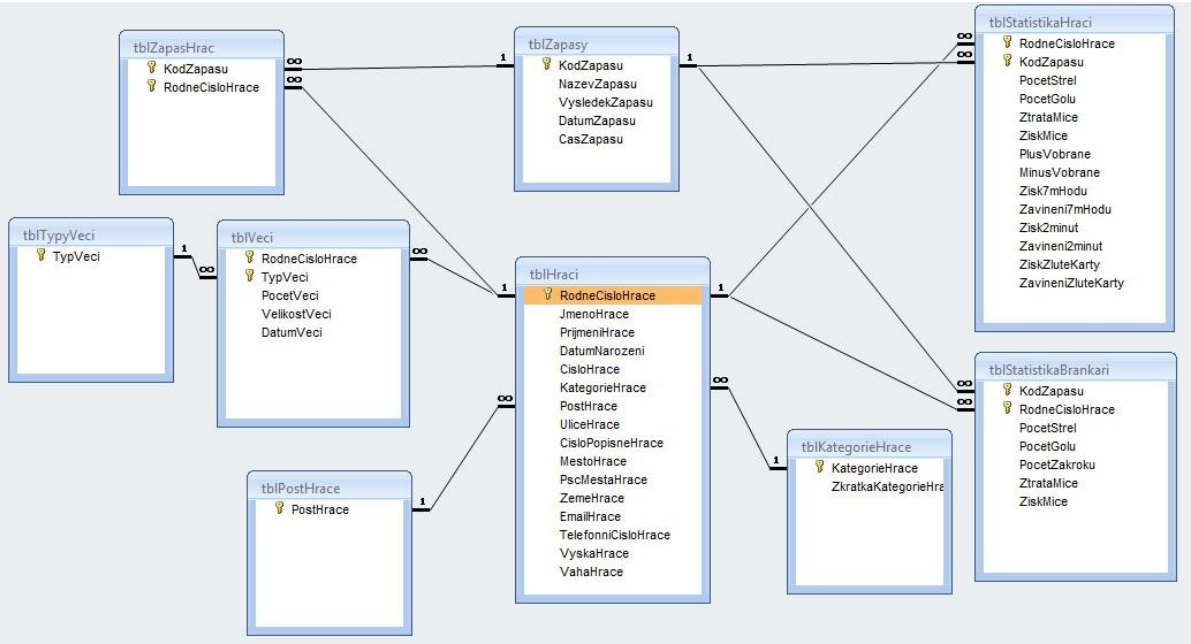
4.1.7 Návrh zdrojových tabulek

V rámci aplikace byly vytvořeny tři tabulky, u kterých jsou jejich záznamy zdrojem řádků polí jiných tabulek. Konkrétně jsou to tabulky *tblKategorieHrace*, *tblPostHrace.* Záznamy v těchto tabulkách odpovídají atributům KategorieHrace a PostHrace v tabulce

tblHraci. Poslední tabulkou je *tblTypyVeci*, kde záznamy odpovídají atributu *TypVec* v tabulce *tblVeci*.

4.2 Vytvoření vztahů

Po vytvoření tabulek a definování jejich vlastnosti bylo potřeba tyto tabulky propojit. Propojení bylo vytvořeno dle vazeb, které odpovídaly jednotlivým tabulkám. Příkladem propojení může být propojení mezi tabulkami *tblHraci* a *tblVeci*. Vazba byla vytvořena ve vztahu 1:N, tedy jednomu hráči lze přiřadit více věcí. Tohoto vztahu bylo docíleno propojením primárního a cizího klíče polem *RodneCisloHrace*. Při vytváření je potřeba nadefinovat vlastnosti každého vztahu, které v tomto případě byly povoleny dvě, tedy zajištění referenční integrity a aktualizace souvisejících polí v kaskádě. Odstranění souvisejících polí v kaskádě bylo nastaveno pouze u vztahu mezi tabulkami *tblPostHrace* a *tblKategorieHrace* s tabulkou *tblHraci*, protože pokud bude odstraněna kategorie, tak bude hráč převeden do jiné kategorie, a pokud bude odstraněn post hráče, tak bude muset být převeden na post jiný. Veškeré vztahy, byly vytvořeny ve formě 1:N. Za zmínku stojí propojení tabulek hráčů a zápasů, kde je pomocí vazební tabulky zajištěn vztah M:N.



Obrázek 4.1: Vytvoření vztahů

4.3 Návrh formulářů

Návrh formulářů se odrážel od domluvy s klubem, takže v barvách formulářů se odrážejí barvy kluby, tedy žlutá a zelená. Pozadí prvků formulářů bylo zvoleno na světle šedou barvu, aby vynikl černý text v popisu prvků a příliš „nebily do očí“ ve spojitosti s pozadím, jak by tomu bylo kupříkladu u bílého pozadí prvků.

4.3.1 Návrh formuláře frmHraci

Ve formuláři *frmHraci* se budou zobrazovat data z tabulky *tblHraci*. Formulář je konstruován tak, že v záhlaví formuláře byl vytvořen panel složený s fotografií hráčů a popisu daného formuláře. Tento panel byl vytvořen v programu Corel Draw a přidán jako obrázek. Dále v rámci těla formuláře byly vytvořeny prvky formuláře, které byly přidány na základě existujících polí a obsahují tedy kompletní atributy z tabulky *tblHraci*.

Po vytvoření a rozmístění prvků formuláře byly vytvořeny tlačítka na přidání, odstranění a úpravu hráčů. Podstatou je, že celý formulář je uzamknutý, tedy do polí se nelze dostat, a tak pomocí těchto tlačítek lze data upravovat v rámci tabulky *tblHraci.*

Tlačítko *Přidat hráče* umožňuje přidání hráče do databáze. Po kliknutí na tlačítko bude otevřen samostatný formulář, který obsahuje nevázaná pole odpovídající polím v tabulce *tblHraci.* Každé pole, bylo ošetřeno proti nepřesným či chybným výrazům, jako je například zadání čísla do příjmení hráče, špatné zadání rodného čísla ve vstupní masce či vybrání kategorie hráče pouze z předem nadefinovaného seznamu. Také bylo zajištěno, aby všechny povinné položky, které jsou označeny hvězdičkou, byly vyplněny, jinak bude zavedeno chybové hlášení. Ve formuláři *Přidání nového hráče* byla vytvořena dvě tlačítka, a to *Přidat* a *Storno.* Po kliknutí na tlačítko *Přidat* je hráč přidán do tabulky *tblHraci*. Pokud však dojde k určité chybě, jako je například shoda rodné čísla hráče v tabulce a aktuálním formuláři *Přidání nového hráče*, tak nastane opět chybové hlášení a kurzor je převeden do pole s chybou v záznamu. Pokud přidání hráče proběhlo úspěšně, je formulář zavřen a vše je vráceno na formulář *frmHraci*. Stejná procedura vrácení je vykonána i při kliknutí na tlačítko *Storno.*

Pomocí tlačítka *Upravit hráče* je otevřen formulář se stejnou strukturou, jako tomu bylo u přidání hráče do databáze, s tím rozdílem, že v polích jsou zobrazeny hodnoty aktuálního hráče zobrazeného ve formuláři *frmHraci* a je možno tyto hodnoty upravovat. Může se stát, že hráč neobsahuje žádnou statistiku. Pro tento případ zde bylo vytvořeno ošetření daného problému a uživatel je opět upozorněn chybovým hlášením. V rámci úprav lze měnit všechny hodnoty až na *Rodné číslo hráče*, protože touto úpravou by byla změněna

identita hráče, což by mohlo vést k chybným záznamům. Záznamy lze upravovat jen pomocí nadefinovaných polí, tedy nejde například zadat číslo do příjmení hráče či vymazání hodnoty tak, že žádná hodnota nebude nadefinovaná, vše opět vede k chybovému hlášení po stisknutí tlačítka *Upravit*. Pokud jsou tedy splněny všechny podmínky úpravy hráče je po stisknutí tlačítka *Upravit* hráč úprava zapsána do tabulky a vše je vráceno na formulář *frmHraci*. Opět se tatáž procedura provede při kliknutí na tlačítko *Storno*.

Obrázek 4.2: Formulář frmHraci

Posledním tlačítkem v rámci provádění úprav hráčů je tlačítko *Odstranit Hráče*. Po kliknutí na tlačítko je vznesen dotaz s hodnotami aktuálního hráče, zda má být hráč odstraněn. Pokud ano, tak je hráč odstraněn, pokud ne, tak je vše vráceno na formulář *frmHraci*.

Jelikož je hráčů v rámci databáze v současném stavu přesně 108 a jsou rozděleni do jednotlivých kategorií je potřeba hráče vyhledávat. K tomuto slouží dvě pole se seznamem.

První z nich má název *VyhledavaniHrace* a zdrojem záznamů zde je tabulka *tblHraci*. Hráče lze vyhledat několika možnostmi, a to pomocí projetí roletky a výběru konkrétního hráče nebo pomocí zadání jeho jména, příjmení, či rodného čísla do pole. K vyhledání dojde při kliknutí na tlačítko *Vyhledat*. Pokud je zadáno do pole se seznamem příjmení tak je pomocí našeptávače automaticky napovězeno příjmení hráče. Pokud po stisknutí tlačítka *Vyhledat* hodnota ve *VyhledavaniHrace* souhlasí s hodnotou v tabulce *tblHraci*, tak je provedeno filtrování pomocí zadané hodnoty. Pokud hodnota nesouhlasí, je vyhozeno

chybové hlášení a kurzor je vrácen zpět do pole se seznamem *VyhledavaniHrace.* Vedle tlačítka *Vyhledat* se nachází tlačítko *Zobrazit vše.* Po jeho stisknutí se vypíná filtrování, a jsou zobrazení všichni hráči.

Druhé vyhledávací pole se nazývá *VyberKategorie.* Principiálně má stejnou filozofii jako předchozí vyhledávání hráčů, ale zde se hráči filtrují pomocí kategorie. Zdrojem řádků je tedy tabulka *tblKategorieHrace.* Vyhledávání je omezeno na seznam, takže zadaná hodnota musí odpovídat hodnotě v seznamu.

V rámci formuláře *frmHraci* byly vytvořeny podsestavy. Tyto podsestavy jsou celkem čtyři a odkazují se na formuláře *frmVeci*, *frmHracZapas*, *frmHracStatistika*, *frmBrankarStatistika.*

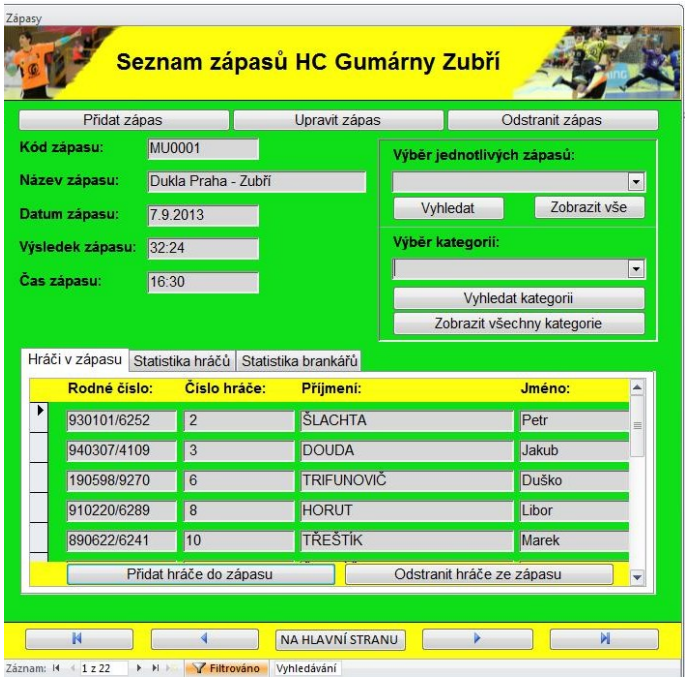
Formuláře *frmHracZapas*, *frmHracStatistika*, *frmBrankarStatistika* jsou dle požadavku klubu určeny pouze pro čtení, a tak je nelze z pohledu formulář hráčů nijak upravovat či měnit. Tato problematika je řešena ve formuláři *frmZapasy.*

Formulář *frmVeci* již přímo souvisí s daným hráčem, a tak jej lze upravovat. Zdrojem řádků je tabulka *tblVeci.* Uspořádání prvků bylo zvoleno jako tabulkové, protože záznamy zde budou řazeny pod sebe v rámci tzv. nekonečného formuláře. Propojení s aktuálním hráčem je zřízeno pomocí propojovacího pole *RodneCisloHrace.* Jak již bylo řečeno lze záznamy v tomto formuláři upravovat, ale opět to nebude přímo ve formuláři, ale opětovně za pomocí tlačítek jako u formuláře *frmHraci.* Jsou zde tedy tři tlačítka s názvem *Přidat záznam*, *Upravit záznam* a *Odstranit Záznam*, které obsluhují záznamy v daném formuláři charakteristicky podobně, jako tomu bylo u *frmHraci* a podle aktuálního záznamu, kterého je docíleno pomocí voliče záznamů, vykonají jimi charakteristickou proceduru v rámci tabulky *tblVeci*. Opět zde může nastat případ, že hráč nebude mít žádný záznam, a tak i zde je ošetřeno kliknutí na tlačítko o upravení či vymazání hráče chybovým hlášením. Ve druhé sadě o třech tlačítkách jsou *Přidat novou věc do seznamu*, *Upravit věc v seznamu, Odstranit věc ze seznamu.* Tyto tlačítka otevírají samostatné formuláře, kde zdrojem řádků je tabulka *tblTypyVeci* a v rámci těchto formulářů jsou prováděny dané procedury v rámci tabulky *tblTypyVeci*. Opět je i zde zajištěno ošetření a zobrazení chybového hlášení při kliknutí na tlačítka *Upravit věc v seznamu, Odstranit věc ze seznamu* u neexistujících záznamů. V rámci kliknutí na tlačítko upravit a následného zobrazení formuláře *frmUpravitVecVseznamu* jsou zamítnuty opravy u primárního klíče, tedy u polí *RodneCisloHrace* a *TypVeci.*

Na závěr formuláře *frmHraci* je v zápatí vytvořena navigační nabídka, pomocí níž lze přecházet mezi záznamy či na hlavní stranu. Součástí formuláře je i klasický volič záznamů, který nabízí aplikace MS Access.

4.3.2 Návrh formuláře frmZapasy

V podobném duchu, ale samozřejmě s jinými hodnotami je vytvořen i formulář frmZapasy. Rozložení prvků ve formuláři, stejně tak jako vyhledávání je přizpůsobeno návrhu formuláře o přehledu hráčů. Tedy zdrojem prvků nebo lépe řečeno polí formuláře jsou existující tabulky *tblHraci*. Součástí formuláře jsou opět tlačítka pro přidávání, upravení a odstranění zápasu, která vycházejí z hodnot aktuálního záznamu. Po stisknutí těchto tlačítek se opět provedou jim charakteristické události aplikované na aktuální záznam formuláře. Rozdílnou vlastností oproti formuláři *frmHraci* je, že při pokusu o vymazání zápasu, který obsahuje hráče, se zobrazí chybové hlášení o zamítnutí vymazání zápasu.



Obrázek 4.3: Formulář frmHraci

Vyhledávání je zajištěno rovněž pomocí dvou vyhledávacích polí se seznamem, kde jedno pole slouží k vyhledávání konkrétního zápasu, kde pomocí zdrojů záznam z tabulky *tblZapasy* je porovnávána hodnota se záznamy ve formuláři a při shodné záznamu dochází k vyhledání, jinak opět k chybovému hlášení. Druhým vyhledávacím polem se seznamem je výběr kategorie, kde pomocí porovnání hodnot ze seznamu, u kterých je zdrojem tabulka *tblKategorieHrace*, v rámci které bylo vytvořeno pole se zkratkami jednotlivých kategorií, které odpovídají prvním dvěma písmenům kódu zápasu, podle kterých je rozeznávaná daná kategorie.

Formulář byl opět řešen tak, že jeho součástí jsou podsestavy. Jsou zde tři, a to *frmZapasHraci*, *frmStatistikaHraci*, *frmStatistikaBrankari.*

Formulář *frmZapasHrac* zobrazuje přiřazení hráče k danému zápasu. Přiřazení k danému zápasu je opět pomocí tlačítka, kde se po stisknutí otevře formulář s uzamknutým kódem zápasu, který odpovídá aktuálnímu záznamu ve formuláři a k tomuto kódu lze přiřadit pouze existujícího hráče, tedy zdrojem záznamů zde je tabulka *tblHraci.*

Ve formulářích *frmStatistikaHraci* a *frmStatistikaBrankari* se filosofie návrhu opět opakuje.

4.3.3 Návrh formuláře frmUvod

Po nastartování aplikace je zobrazen formulář *frmUvod.* Do záhlaví byl přidán název klubu. V rámci těla formuláře byly definovány celkem dvě tlačítka, které odkazují na formuláře aplikace. Následující tři přepínače slouží k výěru jednotlivých sestav. Po vybrání příslušné sestavy je po stisknutí na tlačítka tisk otevřen formulář s výběrem příslušné kategorie. Po vybrání kategorie je otevřena aktuální sestava odpovídající dané kategorii.

Obrázek 4.4: Formulář hlavní strany

4.3.4 Návrh formuláře frmStart

Ke dvěma hlavním formulářů je potřeba se nějakým způsobem dostat. Proto bylo vytvořeno rozhraní před těmito formuláři. Úplně úvodním formulářem, který je zaveden po spuštění aplikace je *frmStart.* Po uvedení aplikace do běhu je otevřen tento úvodní formulář,

obsahují zprávu o přivítání uživatele v aplikaci. Formuláře bude obsahovat pouze logo klubu a po automatickém zavedení se pomocí nastavení časování po třech sekundách přepne na hlavní stranu aplikace.

4.4 Naplnění databáze daty

Po vytvoření struktury a nadefinování všech událostních procedur bylo potřeba databázi naplnit daty. Zde byl vznesen požadavek klubu o vytvoření hráčů, přiřazení hráčů k zápasům, přiřazení každému hráči věci, ale v rámci zaučení a seznámení se s aplikací nevyplňovat statistiky k jednotlivým hráčům. V rámci zajištění a důkazu funkčnosti databáze, tak byly přidány jen určité statistiky, které budou při předávání aplikace klubu smazány.

4.5 Dotazy

Dotazy jsou vytvářeny za účelem spojování dat z tabulek. V rámci aplikace byly vytvořeny celkově tři dotazy. Ovšem další dotazy jsou zakomponovány v rámci událostních procedur jazyka VBA jako je například odstraňování hráčů z databáze apod.

Prvním druhem je dotaz *BrankariAzapasy.* Tento dotaz je tvořen potřebnými poli z tabulek *tblHraci, tblZapasy* a *tblStatistikaBrankari*. Následující dotaz je *HraciAzapasy*. Pole jsou do dotazu získávána z tabulek *tblHraci*, *tblZapasy* a *tblStatistikaHraci*. Posledním druhem dotazu má název *HraciAveci* se zdrojovými tabulkami *tblHraci* a *tblVeci*.

Tyto dotazy jsou zdroji pro reprodukci dat.

4.6 Návrh sestav

Tisk dat a jejich zobrazování je potřebným výstupem každé databáze. I tato aplikace obsahuje reprodukci dat pomocí sestav. Sestavy byly vytvořeny celkem tři.

4.6.1 Návrh sestav hráčů a jejich věcí

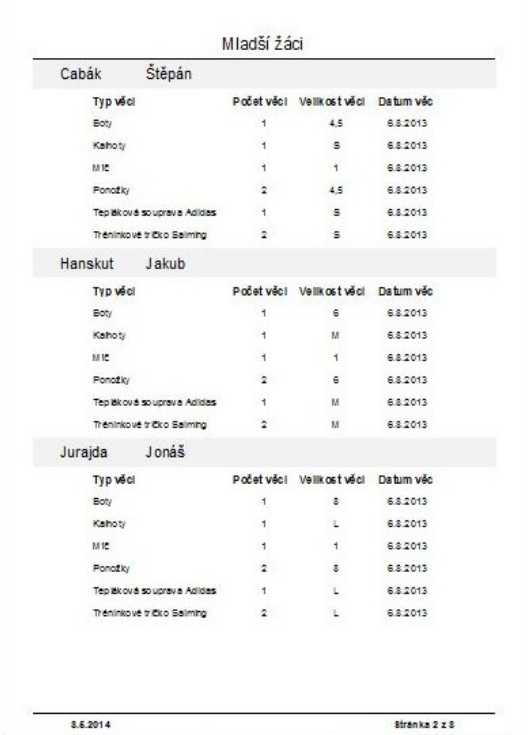
Každá sestava je složena s různých záhlaví, těla a zápatí. V rámci této sestavy bylo záhlaví stránky vytvořeno jako nadpis stránky, obsahují seskupení podle kategorie hráčů obsažených v sestavě. Nadpis byl nadefinován pouze na první stranu.

Následující záhlaví obsahuje příjmení a jméno hráče určující další úroveň seskupení. V rámci tohoto záhlaví byly vytvořeny i popisky jednotlivých polí v rámci tabulkového rozložení prvků formuláře.

Samotné tělo sestavy obsahuje záznamy o jednotlivých věcech, kterými disponuje daný hráč. Zdrojem sestavy je dotaz *HraciAveci* s patřičnou zkratkou charakterizující kategorii. Nadefinovaná byla i vlastnost, že každý záznam hráče bude zobrazen kompletně na

jedné straně, tedy je spjat s daným sjednocení na každé straně zvlášť. Je tedy zajištěno, že nelze dojít k rozložení záznamů každého hráče přes dvě strany.

Pro zajištění přehledu o sestavě bylo v zápatí stránky vytvořeno dle agregačních funkcí číslování stránek a zobrazení aktuálního data tisku sestavy.


Mladší žáci

Cabák Štěpán

Typ věci	Počet věcí	Velikost věcí	Datum věc
Boty	1	4,5	6.8.2013
Kalhoty	1	S	6.8.2013
Mič	1	1	6.8.2013
Ponožky	2	4,5	6.8.2013
Tepláková souprava Adidas	1	S	6.8.2013
Tréninkové tričko Salming	2	S	6.8.2013

Hanskut Jakub

Typ věci	Počet věcí	Velikost věcí	Datum věc
Boty	1	6	6.8.2013
Kalhoty	1	M	6.8.2013
Mič	1	1	6.8.2013
Ponožky	2	6	6.8.2013
Tepláková souprava Adidas	1	M	6.8.2013
Tréninkové tričko Salming	2	M	6.8.2013

Jurajda Jonáš

Typ věci	Počet věcí	Velikost věcí	Datum věc
Boty	1	8	6.8.2013
Kalhoty	1	L	6.8.2013
Mič	1	1	6.8.2013
Ponožky	2	8	6.8.2013
Tepláková souprava Adidas	1	L	6.8.2013
Tréninkové tričko Salming	2	L	6.8.2013

8.8.2014 Stránka 2 z 8

Obrázek 4.5: Sestava hráčů a věcí

Ke každé sestavě byla vytvořena i jednoduchá úvodní strana s názvem daném sestavy a aktuálním datem tisku sestavy sloužící pro přehled a jednoduchém vyhledávání sestavy například po vytisknutí a uložení do kartotéky.

4.6.2 Návrh sestav hráčů a jejich statistik

Převzetí struktury z předchozí sestavy je základem i pro druhý typ sestavy. Rozdílem je zde zdroje řádků, který je zajištěn pomocí dotazu ze tří tabulek *tblHraci*, *tblZapasy* a *tblStatistikaHraci.* Logicky s tím souvisí i pole sestavy související s dotazem.

Záhlaví stránky je neměnné, a tak zde figuruje seskupení dle dané kategorie. Následující seskupení v hierarchii je dle kódu zápasu, kterému odpovídá název zápasu, výsledek, datum a čas odehrání zápasu. V rámci tohoto seskupení byly opět vytvořeny popisky charakterizující pole statistik hráčů. Tedy každý záznam obsahuje příjmení hráče, jméno hráče, číslo hráče a jednotlivé statistiky hráče.

V rámci tohoto reportu vedení klubu požadovalo součty technických chyb v rámci

zápasu, statistiku procentuální úspěšnosti střelby a počet minut v oslabení. Tohoto je docíleno pomocí jednoduchých agregačních funkcí v rámci zápatí spojené s kódem zápasu.

4.6.3 Návrh sestav brankářů a jejich statistik

Statistiky hráčů jsou mírně odlišné od statistika brankářů. Proto i pro brankáře jsou vytvořeny zvláštní sestavy. Nicméně charakteristika zůstávám opět téměř totožná. Zdrojem řádků jsou stejné jako u hráčů, ale místo tabulky se statistikou hráčů zde je soudržnost s tabulkou *tblStatistikaBrankari*.

Statistiku, kterou chce vedení klubu sledovat v rámci každého zápasu této sestavy je opět počet technických a procentuální statistika úspěšnosti zákroků. Statistika dvou minutových trestů u brankářů dochází zcela výjimečně, a tak po konzultaci s klubem není pole o dvou minutových trestech ani součástí tabulek statistik brankářů, protože v naprosté většině případů by obsahovala nulovou či prázdnou hodnotu.

4.6.4 Zobrazování sestav

K zobrazování sestav je využito hlavní stránky aplikace. Na této stránce byly vytvořeny tři přepínače, které odpovídají jednotlivým sestavám. Po výběru jednotlivé sestavy je po kliknutí na tlačítko *Tisk* otevřen formulář s výběrem kategorie, podle které bude filtrováno. Tato vlastnost byla vytvořena na požadavek sekretariátu klubu, kde jsou seznamy tisknuty v rámci kategorií. Pokud je vybrána daná kategorie, tak pomocí tlačítka OK je otevřena sestava v náhledu, poté je možno sestavu vytisknout.

4.7 Shrnutí fungování aplikace

Každá aplikace má určitý sled událostí a kontextový rámec aplikace. Proto i zde pro upřesnění byl vytvořen popis aplikace.

V této aplikaci je po zavedení otevřen úvodní formulář, který uživatele přivítá v aplikaci. Po uplynutí tří sekund se automaticky přepíná na hlavní stránku obsahující dvě tlačítka sloužící pro přechod do databáze kde je možný přehled, úpravy, přidávání či odstranění v jednom případě hráčů a jejich věcí, ve druhém případě přehledu zápasů, hráčů v zápasech a jejich statistik. Po ukončení různých procesů je možno se vrátit zpět na hlavní stranu.

Hlavní strana obsahuje dále tři přepínací tlačítka sloužící pro výběr jednotlivých sestav po výběru jednotlivé sestavy a stisknutí tlačítka *Tisk* je zobrazen formulář s výběrem jednotlivé kategorie, pro kterou je potřeba sestavu tisknout. Po výběru kategorie je zobrazen náhled sestavy.

4.8 Uzavření databáze

Ne každý uživatel je znalý v aplikaci MS Access a tak, ale i z důvodu přehlednosti, zabezpečení úpravy aplikace a klidného procházení aplikace bez nadbytečných prvků, je potřeba aplikaci uzavřít a omezit prvky nesouvisející se samotným chodem aplikace.

4.8.1 Skrytí standardního pásu karet a vytvoření vlastního pásu karet

Databázová aplikace umožňuje skrytí standardního pásu karet. Tato možnost lze provést jednoduše v nastavení možnosti aktuální databáze, kde lze zamítnout potřebné položky tak, aby zůstala pouze karta *Domů*. V rámci těchto možností byly skryty všechny pásy karet, přístup do návrhového zobrazení a vypnutí navigačních podoken.

Mnohem výhodnější je však vytvoření vlastního pásu karet. Dle domluvy a požadavků klubu byl vytvořen vlastní pás karet, který obsahuje pouze ukončení aplikace, a protože bude zde nutná práce se sestavami, tak byl pás karet doplněn i o prvky související se sestavou.

Pás karet byl vytvořen pomocí systémové tabulky *USysRibbons*, která obsahuje tři pole, a to *ID* definováno jako primární klíč s datovým typem automatické číslo, pole *RibbonName*, které je definováno jako datový typ *TEXT* a slouží k pojmenování pole s XML kódem. Název tohoto záznamu v tomto poli je *Ribbon_Konec* je posledním polem je *RibbonXml* definované jako datový typ *MEMO* a v rámci tohoto pole je definovaný již zmiňovaný kód XML. Po restartování aplikace bylo vybráno v možnostech aplikace v nastavení pásů karet a nástrojů ze systémové tabulky pole *Ribbon_Konec* čímž se aktivoval nový pás karet.

ID	RibbonNam	RibbonXml
1	Ribbon_Konec	<customUI xmlns="http://schemas.microsoft.com/office/2006/01/customui"> <ribbon startFromScratch="true"> <tabs> <tab id="idAplikace" label="Ukončení aplikace" visible="true"> <group id="idSoubor" label="Ukončení"> <control idMso="FileExit" label="Konec" enabled="true"/> </group> <group id="idSestavy" label="Sestavy"> <control idMso="FilePrintPreview" label="Náhled" enabled="true"/> <control idMso="PrintDialogAccess" label="Tisk" enabled="true"/> <control idMso="PrintPreviewClose" label="zavřít" enabled="true"/ </group> </tab>

Obrázek 4.6: Vytvoření vlastního pásu karet [10]

4.8.2 Zabezpečení databáze pomocí hesla

Dalším základním kamenem databáze je její zabezpečení. Již bylo vytvořeno jakési částečné zabezpečení proti úpravám databáze, nicméně nebylo zajištěno přihlašování do databáze, pokud by chtěl někdo neadekvátně tuto aplikaci otevřít. Možnost zabezpečení

pomocí hesla lze v Accessu jednoduše vytvořit, a proto bylo využito i této možnosti. Nejprve je potřeba databázi otevřít s tzv. výhradním přístupem. Toho lze docílit pomocí karty *Otevřít*, kde byl pomocí výběrové nabídky vybrán aktuální databázový soubor. Po otevření tohoto souboru v Accessu je na kartě soubor nabídka zašifrovat databázi pomocí hesla. Bylo tedy pomocí této nabídky zadáno heslo databáze, které při otevření bude zadáno pro její zpřístupnění.

4.9 Implementace aplikace

Po odevzdání bakalářské práce bude aplikace zaznamenána na paměťové médium a odevzdána do klubu k jejímu využívání. S implementací by neměl být žádný problém, jelikož pracoviště v klubu disponuje výkonným počítačem, jehož součástí je i kancelářský balík MS Office 2010 Professional Plus 2010.

5 Závěr

Po konzultaci a náhodném zjištění poněkud nevyhovujícího zpracovávání dat v klubu HC Zubří, byla učiněna domluva se sekretariátem klubu o návrhu vylepšení stávající situace. Výsledkem bylo zadání databázové aplikace. Konstrukce práce měla být stanoven tak aby zahrnovala přehled hráčů, jejich věcí vypůjčených v rámci klubu, zainteresování hráčů do jednotlivých zápasů a vytvoření statistik k těmto zápasům. Postup práce, odvíjející se od stanovených cílů lze sdružit do tří fází.

Cílů bylo dosaženo následovně:

- získání osobních informací o jednotlivých hráčích pomocí vyplnění předem vytvořených formulářů, zpracování těchto dat a zainteresování do databáze,
- vytvoření databázové aplikace pomocí MS Access, obsahující hráče, jejich přehled věci a jednotlivé zápasy se statistikami,
- částečné splnění cíle bylo zlepšení u reprodukce dat o výkonosti a přehledu hráčů, nicméně tento cíl není fakticky podložen, protože databáze v současném stavu není nasazena do provozu v klubu, takže tento cíl je spíše než splněn od databáze očekáván.

Získání osobních informací, které byly mimo jiné v klubu neúplné nebo dokonce i chybějící vycházelo z osobních styků a je zaručena naprostá přesnost těchto dat.

Konstrukce aplikace vycházela z vytvoření základních tabulek, nadefinování tabulkám patřičné vlastnosti, propojení dle relací, vytvoření formulářů z důvodu přijatelnější podoby prohlížení dat a propojení formulářů pomocí událostních procedur jazyka Visual Basic for Application.

Závěrečná část vytváření aplikace se skládala z vytvoření sestav pro reprezentaci výstupů z databáze a uzavření databáze pomocí skrytí panelů a vytvoření hesla pro bezpečnost aplikace proti útokům.

V současné době je aplikace po funkční stránce plně připravena k nasazení do provozu. Lze jednoduše a účelně přidávat hráče, vytvářet jim statistiky v rámci zápasu či každý záznam intuitivně upravit. Při zadávání požadavků byla s klubem vytvořena dohoda, týkající se pomoci při případných komplikacích, ale hlavně při vylepšení aplikace o další funkce. Je tedy přislíbená možnost konzultace nebo úplné opravy či vylepšení aplikace.

Do budoucna by se dala databáze rozšířit o kontrolu docházky na tréninky, nicméně k tomuto kroku v současné době nechtělo vedení klubu přikročit, paradoxně z důvodu zvyklosti a v současné době určité nepotřebnosti. Nepotřebnost byla argumentována časovou komplikovaností, při každodenní zadáváním dat, a také argumentem o postačujících vlastnostech tabulkového procesoru.

Tato aplikace byla první zkušeností ve vytváření aplikace na základě podmínek a představ druhé strany. Podmínky klubu zněly, aby byla aplikace účelná, intuitivní, charakteristická designem pro klub a jednoduchá na ovládání. Všechny tyto podmínky byly splněny. Práce byla obrovským přínosem a především získání zkušenosti, jak v organizaci práce, tak v komunikaci s druhou stranou a získávání a zpracování informací.

Kontext práce vycházel z nadefinované struktury dle zadání bakalářské práce a všechny části byly v rámci této práce popsány.

Seznam použité literatury

Literatura:

[1] CONOLLY, T., C. BEGG a R. HOLOWCZAK. *Mistrovství - databáze: profesionální průvodce tvorbou efektivních databází.* Brno: Computer Press, 2009. ISBN 978-80-251-2328-7.

[2] HERNANDEZ, Michael J. *Návrh databází: podrobný průvodce*. Praha: Grada, 2006. ISBN 80-247-0900-7.

[3] KRUCZEK, Aleš. *1001 Tipů a triků pro Microsoft Access 2007/2010.* Brno: Computer Press, 2011. ISBN 978-80-251-3507-5.

[4] KRUCZEK, Aleš. *Microsoft Access 2010: podrobná uživatelská příručka*. Brno: Computer Press, 2010. ISBN 978-80-251-3289-0.

[5] OPPEL, Andrew. *Databáze bez předchozích znalostí.* Brno: Computer Press, 2006. ISBN 80-251-1199-7.

[6] PÍSEK, Slavoj. *Access 2010: podrobný průvodce.* Praha: Grada, 2011. ISBN 978-80-247-3653-2.

[7] SHEPHERD, Richard. *Access VBA: výukový kurz.* Brno: Computer Press, 2012. ISBN 978-80-251-3686-7.

[8] VOGLOVÁ, Blanka. *Acces v kanceláři - typické činnosti krok za krokem*. Praha: Grada Publishing, 2002. ISBN 80-247-0321-1.

Internetové zdroje:

[9] MARČEK, Dušan. *Přednášky předmětu: Základy databázových systémů B* [online]. 2013. [cit. 2014-03-01]. Dostupné z: http://lms.vsb.cz/mod/resource/view.php?id=4074.

[10] NOVÁK, Vítězslav. *Přednášky předmětu: Databázové aplikace* [online]. 2012. [cit. 2014-02-26]. Dostupné z: http://lms.vsb.cz/mod/resource/view.php?id=22278.

Seznam zkratek

1NF - První normální forma

2NF - Druhá normální forma

3NF - Třetí normální forma

4NF - Čtvrtá normální forma

5NF - Pátá normální forma

BCNF - Boyce Coddova normální forma

HC – Handball Club

IBM - International Business Machines Corporation

IME - Input Method Editor

IT – Information Technology

MS - Microsoft

MŠMT – Ministerstvo školství, mládeže a tělovýchovy

SCM – Sportovní centrum mládeže

SQL - Structured Query Language

SŘBD – Systém řízení báze dat

VBA – Visual Basic for Application

VBE – Visual Basic Editor

Prohlášení o využití výsledků bakalářské práce

Prohlašuji, že

- jsem byl seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. – autorský zákon, zejména § 35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a § 60 – školní dílo;

- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen VŠB-TUO) má právo nevýdělečně, ke své vnitřní potřebě, bakalářskou práci užít (§ 35 odst. 3);

- souhlasím s tím, že bakalářská práce bude v elektronické podobě archivována v Ústřední knihovně VŠB-TUO a jeden výtisku bude uložen u vedoucího bakalářské práce. Souhlasím s tím, že bibliografické údaje o bakalářské práci budou zveřejněny v informačním systému VŠB-TUO;

- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu § 12 odst. 4 autorského zákona;

- bylo sjednáno, že užít své dílo, bakalářskou práci, nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat příměrný příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich výše).

V Ostravě dne 9. 5. 2014

……

Tomáš Janků

Seznam příloh

Příloha č. 1: CD (soubor) s databázovou aplikací